NASA TECHNICAL NOTE

NASA TN D-4729

C.1

NASA TN D-4729

A TRAJECTORY CODE FOR MAXIMIZING THE PAYLOAD OF MULTISTAGE LAUNCH VEHICLES

by Omer F. Spurlock and Fred Teren

Lewis Research Center
Cleveland, Ohio

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION • WASHINGTON, D. C. • AUGUST 1968

A TRAJECTORY CODE FOR MAXIMIZING THE PAYLOAD OF MULTISTAGE LAUNCH VEHICLES

By Omer F. Spurlock and Fred Teren

Lewis Research Center
Cleveland, Ohio

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION

For sale by the Clearinghouse for Federal Scientific and Technical Information
Springfield, Virginia 22151 – CFSTI price $3.00

ABSTRACT

A computer code is described that quickly maximizes the payload of a multistage launch vehicle. Calculus of variations techniques are used to optimize the trajectory and propellant loadings (optional) of one to six stages. The flow rate and vacuum thrust of each stage are constant. The jettison weight is a linear function of the propellant loading. The computer time is greatly reduced by minimizing the integration of atmospheric (booster) portions of the trajectory. A table of booster burnout conditions is formed from a limited number of boost trajectories. The table is used in conjunction with an interpolation scheme to obtain initial conditions for the vacuum portion of the trajectory.

CONTENTS

A TRAJECTORY CODE FOR MAXIMIZING THE PAYLOAD OF MULTISTAGE LAUNCH VEHICLES

by Omer F. Spurlock and Fred Teren

Lewis Research Center

SUMMARY

The computer code described in this report quickly optimizes the trajectory in order to maximize the payload of a multistage launch vehicle where the propellant loading of one or more of the stages has not been fixed. The flow rate and vacuum thrust of each of a maximum of six stages are constant. The jettison weight for each stage is a linear function of the propellant weight of the stage. The steering profile and the optimum stage size criteria are determined by a calculus of variations solution. The program will optimize trajectories for several types of final conditions. The program is coded in FORTRAN IV and is currently running on a direct couple system (IBM 7094-II/7044).

INTRODUCTION

The problem of maximizing the payload of a multistage launch vehicle is difficult when the stage sizes as well as the trajectory must be optimized. In reality, the thrusts, flow rates, hardware weights, and steering profile are complicated functions of many variables. Maximizing the payload of a vehicle by using a mathematical model that incorporates such functions would be a formidable and time-consuming task. However, the complicated model may usually be replaced by a much simpler one (as is done in this code) without greatly affecting the significance of the results. In fact, for new vehicles, the vehicle characteristics are not known in enough detail to allow use of a complicated model.

A code was developed in response to the need for quick maximization of the payload capability of launch vehicles in preliminary mission studies. The code will optimize the trajectory and propellant loadings of a launch vehicle with a maximum of six stages. Fixed or optimized coast phases are permitted by treating them as stages. The mathematical model assumed for the analysis used by the program has many simplifications.

The vacuum thrust and flow rate of each stage are constant, and the hardware weights are linear functions of the stage propellant loadings. The trajectories are planar with the atmospheric portion being integrated in three dimensions and later or upper stages being integrated in two. The booster stage is that portion of the trajectory which is in the atmosphere and is constrained by aerodynamic loads and heating to a near-zero angle-of-attack trajectory (explained in the BOOSTER section). The upper-stage steering profile, as well as optimum staging criteria, is determined by a calculus of variations analysis. This analysis is described in reference 1.

In order to avoid confusion, a distinction is made at this point between stages of a vehicle and what shall be called the phases of a trajectory. The term stage shall refer to the actual hardware divisions of a vehicle, and phase shall be used to designate the parts of a trajectory. It should be understood, for example, that the first stage will operate initially in the atmosphere where the trajectory is constrained, whereas later in the flight the first stage may be out of the atmosphere where the trajectory is unconstrained. If this is the case, it is necessary, for computational purposes, to consider the first stage as two phases: a booster phase and an upper phase.

The booster trajectories are characterized by a short vertical rise of fixed duration after which the vehicle is tilted or kicked over instantaneously at some small predetermined angle in the desired azimuth direction. Thereafter, the steering profile is constrained to yield essentially zero angle of attack until the end of the booster phase.

Models for an oblate earth, atmosphere, and vehicle drag are used. The integration of the booster (aerodynamic) phase characteristically consumes much of the computer time for a problem. Because over this phase of the trajectory only two parameters, kick angle and duration, affect the trajectory, it is practical and advantageous to construct a table lookup scheme in which the booster burnout conditions may be obtained as functions of the two variables. Only those booster trajectories required to furnish points for the curve fit are integrated, thus minimizing the computer time consumed by booster integration. As new points are required, the table expands to include the new trajectories.

Other codes and techniques are available which make different simplifications. Some of these are listed in the references of reference 1. The PRESTO code of reference 2 allows consideration of a more complex problem. The simplifications are less constraining. The trajectories considered are not necessarily planar; the thrusts and flows are not necessarily constants; and coast phases may be inserted between four powered stages. Aerodynamic lift as well as drag forces are included in the computations. The optimization technique is the method of steepest descents. However, PRESTO requires of the order of 1 minute to solve the same problem that would require about 1/10 minute for this code.

The first part of this report contains the assumptions and capabilities of the program

followed by a discussion of the organization. Next, a complete step-by-step set of directions for utilizing the deck is provided. The appendixes contain the FORTRAN listings and related information.

ASSUMPTIONS AND CAPABILITY

The analysis on which the equations in the code are based is presented in reference 1. The set of assumptions for this program are that

(1) The launch weight is fixed.

(2) Vacuum thrusts for all phases are constants.

(3) Flow rates for all phases are constants.

(4) The hardware weight for each phase is assumed to be a linear function of the phase propellant loading defined by

$$W_s = W_H + kW_p \tag{1}$$

where W_s is the total structural weight, W_H is the fixed weight, W_p is the phase propellant weight, and k is the propellant sensitive structure factor. (All symbols are defined in appendix A.)

(5) During the booster phase, drag = $C_D q A_{ref}$, where C_D is a drag coefficient found as a function of Mach number, q is dynamic pressure, and A_{ref} is the reference area. After the booster phase, drag is equal to zero.

(6) Thrust during booster phases is defined by

$$F = F_v - A_e P \tag{2}$$

where F is the thrust, F_v is the vacuum thrust, A_e is the engine exit area, and P is the atmospheric pressure. During upper phases, $F = F_v$.

(7) During the booster portion of the trajectories, the steering profile is determined such that essentially zero angle of attack is maintained. In the upper-phase portion, the steering profile is determined by the calculus of variations.

(8) The upper-phase trajectories are planar, and the thrust vector during the booster portion is constrained to be parallel to the launch azimuth plane.

In addition to the integrated trajectory, provision is made for adding an additional impulsive velocity increment V_I after the desired final conditions have been achieved. The final phase is used to provide this additional velocity increment, and the standard impulsive velocity equations are used to calculate the propellant required for the maneu-

ver. Postorbital maneuvers may often be conveniently simulated as an impulsive velocity increment.

In addition to the preceding set of assumptions, other characteristics of the code are results of the fact that it was constructed by combining two existing codes. The booster portion is a derivative of the N-Body Code of reference 3; the upper-phase portion was constructed from a two-dimensional calculus of variations code designed for flight in a vacuum. This accounts for the booster portion being three dimensional and the upper-phase portion being two dimensional.

The steering program during the booster portion is constrained such that for a short, specified period the vehicle rises vertically and then is instantaneously tilted or kicked over in the desired azimuth direction at some elevation angle henceforth called the kick angle. After the kick maneuver is performed, the angle of attack in the pitch plane is constrained to zero. The booster phase may be composed of six distinct segments each of which is limited to a fixed vacuum thrust, engine exit area, and flow rate. A fixed weight may be jettisoned at the end of each booster segment. The hardware weight for the last booster phase is treated as in the upper phases.

A maximum value may be placed on the propellant loading of any phase. If the optimum loading exceeds the prescribed value, the duration of the phase is fixed such that the maximum loading is used; and the problem is reoptimized.

ORGANIZATION

This code was constructed by combining two existing codes: a booster code and an upper-phase code. This program remains largely divided into these almost independent sections because the two parent codes have different coordinate systems. The booster portion is simple and straightforward in spite of the more complex vehicle model. The upper-phase portion is more complex because it requires a targeting or iteration scheme as well as the calculus-of-variations steering control and staging criteria. Therefore, the discussion of the organization of the program may be conveniently divided into sections on the booster and upper phases with auxiliary sections on the targeting procedure, the final boundary conditions, and input and output. Most of the discussion of the logic which connects the two portions is contained in the booster discussion.

BOOSTER

The booster portion of the trajectory is integrated in a three-dimensional rectangular equatorial coordinate system utilizing the integration scheme, oblate earth model,

and aerodynamic simulation given in reference 3. The method and organization of the booster portion of this code are derived from and, hence, are very similar to reference 3. The atmospheric model was obtained from reference 4.

The boost trajectory is characterized by a short vertical rise of fixed duration after which the vehicle is tilted or kicked over instantaneously at some small predetermined angle in the desired azimuth direction. During the vertical rise, the thrust vector is alined along the launch radius vector. At the end of the vertical rise period, the relative velocity vector is set to the desired kick angle (measured from the horizontal) in the desired azimuth direction. For the remainder of the booster portion of the trajectory, the thrust vector is directed so that the angle of attack in the pitch plane is zero. For further and more extensive discussion of this procedure, see appendix B.

Thus, for a fixed launch weight, booster propulsion characteristics, and launch azimuth, the vertical rise phases are identical; the only changes in the boost trajectories are the result of changes in the kick angle and the duration of the boost phase.

The program is constructed so that the upper-phase portion requests the burnout conditions at the end of the booster phase as a function of the booster kick angle and booster burning time: the two parameters that remain available for optimization in the booster. These burnout conditions (altitude, vertical velocity, and horizontal velocity) could be obtained by integrating the booster trajectory with the precise kick angle and burning time required. This procedure is followed whenever the kick angle and booster phase duration are fixed. However, if either the kick angle or booster phase duration are to be optimized, a table lookup technique is employed. The short vertical rise is integrated only once, and the weight and state conditions are stored. A grid of kick angles α_k, is established with a preselected grid spacing (i.e., . . ., 88.4°, 88.5°, 88.6°, . . ., 89.8°, 89.9°). The grid spacing $\Delta\alpha_k$ is constrained to values such that

$$\Delta\alpha_k = \frac{0.1^\circ}{2^i} \qquad i = 0, 1, 2, \ldots \tag{3}$$

where i is chosen at execution time. This constraint is placed on the grid spacing in order to simplify the logic associated with choosing the kick angles needed to form a satisfactory table. The three kick angles from the grid closest to the desired kick angle are chosen, and trajectories are run using these three kick angles. As the integration proceeds, the state variables (altitude, horizontal and vertical velocities) are stored beginning at some prespecified time and at fixed increments of time until another prespecified time is reached. The upper-phase state variables are then expressed as second-degree polynomials in kick angle and booster-phase duration, and the desired state variables are calculated from these polynomials. The polynomials are of the form

$$y = a_1 + a_2\alpha_k + a_3t_B + a_4\alpha_k t_B + a_5\alpha_k^2 + a_6t_B^2 \tag{4}$$

where y is the dependent state variable, α_k is the kick angle, t_B is the phase duration, and a_1 to a_6 are constants which are evaluated from the stored data. The upper-phase portion of the deck requires not only the burnout conditions already noted, but also requires partial derivatives of the burnout conditions with respect to kick angle and booster-phase duration when optimization of either of these parameters is desired. The partial derivatives are obtained by differentiation of the polynomials. The interpolation scheme requires trajectories for three kick angles from the grid to construct the polynomials. As targeting proceeds, the targeting scheme may request burnout conditions for a kick angle which is not enveloped by previously integrated trajectories. If the required kick angle is less than one grid spacing $\Delta\alpha_k$ from the closest one already available, the polynomials are formed with the three closest kick angles. If the required kick angle is further than $\Delta\alpha_k$ from the nearest available kick angle, trajectories are integrated for one or more kick angles on the grid, such that the closest is within $\Delta\alpha_k$ and the most remote is within $3\Delta\alpha_k$. In this manner, the table is expanded until a range of kick angles covering the area of interest is obtained and stored, thus eliminating the need for reintegrating the time-consuming booster trajectories. This table may be used for all problems sharing the same booster configuration.

When a problem is completed, the booster table may be punched on binary cards, which may be read back into the computer at a later date. If it is desired to investigate a vehicle whose booster configuration is identical to one already considered and for which a booster binary table has been acquired, the binary cards may be reintroduced at execution time, thus avoiding reintegration of the booster trajectories.

UPPER PHASES

At the end of the booster phase, the number of variables free for optimization increases. The booster phase had only two: the kick angle and the booster-phase duration. The upper phases add the steering profile and additional phase durations to the variables that may be optimized. The variational analysis from reference 1 requires the simultaneous integration of the equations of motion and the Euler-Lagrange equations. Additionally, the analysis demonstrates that for an N-phase problem there exist $N + 5$ final conditions to be satisfied and $N + 5$ initial conditions with which to satisfy them. Obviously, for a multistage vehicle the problem of optimizing the phase durations would involve a formidable iteration process. However, the analysis goes on to show how the number of iteration variables may be decreased by resorting to "internal" optimization of some of the parameters which would otherwise be in the "external" iteration loop.

By utilizing the techniques for internal optimization given in reference 1 and by choosing to limit the number of parameters in the external iteration ti five (thus limiting the final conditions to be satisfied to five), a maximum of five upper phases and the booster phase may be optimized. The program automatically utilizes the correct procedure for reducing the iteration loop size. A description of this process is presented in appendix C. The choice of appropriate equations or techniques is a function of which parameters are to be optimized, whether a phase is powered or coasting, the sequential order of the fixed and optimized phases, and whether the structure factor is zero or nonzero.

The initial conditions for the two-point boundary value problem are as follows:

$$
\begin{array}{ll}
\dot{\psi} & \text{initial time rate of change of thrust angle} \\
\psi & \text{initial thrust angle measured with respect to local horizontal} \\
\alpha_k & \text{kick angle for booster} \\
\lambda_4 & \text{travel angle Lagrange multiplier} \\
\tau_i & \text{phase duration, where } i = 1, 6
\end{array}
\tag{5}
$$

The initial rate of change of thrust angle $\dot{\psi}$ is always required as an initial condition. The variables ψ and/or α_k may be required, depending on the problem. The phase durations τ must also be supplied by the user. If a phase is of fixed duration, that value will, of course, determine the duration. If a phase is to be optimized, an initial estimate of the phase duration is required which may be used in the two-point boundary value problem. All these variables must be loaded by the user; the program will use them as required.

As shown in reference 1, λ_4 is a constant during a trajectory and is equal to zero except for nonoptimum travel angle trajectories. This option (nonoptimum travel angles) is seldom interesting in the type of problem for which this program was designed. Furthermore, experience has shown that the difficulties in converging a trajectory with a nonoptimum travel angle are formidable; but for the sake of completeness and the possibility of future interest in that type of problem, it has been included in the Euler-Lagrange equations. It is not included in the iteration loop, and the travel angle is never specified as a final condition.

The upper phases are integrated in a polar coordinate system which assumes a spherical earth. Thrust and gravity are the only forces acting on the vehicle, thus facilitating rapid integration. The upper phase portion of the deck shares the integration technique of the booster portion, which is the fourth order Runge-Kutta scheme described in reference 3.

To recapitulate, the upper phases consist of a maximum of five phases and a minimum of one phase. Each phase has a constant thrust and flow rate. Any upper phase except the final phase may be a coast phase (no thrust or flow rate). Any phase may

have a fixed or optimized duration. The structure weight for each phase including booster phase is assumed to be a function of the propellant expended during that phase (defined by eq. (1)).

In addition to the integrated trajectory, provision is also made for adding an impulsive velocity increment V_I after the desired final conditions have been achieved in the integrated trajectory. The final phase provides the velocity increment.

ITERATION PROCEDURE

The two-point boundary value problem implicit in the solution of the problem is solved by a multivariable Newton-Raphson iteration scheme. The program employs first-order, finite-difference equations to relate changes in final conditions to changes in initial conditions

$$\delta y = M \delta x \tag{6}$$

where δx and δy are n-vectors (with an $n \times n$ iteration assumed: $2 \leq n \leq 5$) denoting differences in initial and final conditions, and M is an $n \times n$ matrix of partial derivatives of final conditions with respect to initial conditions

$$M_{jk} = \frac{\delta y_j}{\delta x_k} \approx \frac{\Delta y_j}{\Delta x_k} \tag{7}$$

The program obtains the matrix M by integrating a reference trajectory and n independent perturbed trajectories (or, as dicussed later, by considering $n + 1$ independent trajectories) so that

$$\Delta y = M \, \Delta x \tag{8}$$

where Δx is an $n \times n$ matrix of differences in initial conditions such that Δx_{jk} is the difference of the j^{th} initial condition on the k^{th} perturbed trajectory and Δy is an equivalent matrix in final conditions. The guesses at the initial conditions are improved by the equation

$$\delta y_r = M \delta x_r \tag{9}$$

where the subscript r indicates differences between reference and desired values; hence,

$$\delta x_r = \Delta x \, \Delta y^{-1} \delta y_r \tag{10}$$

A large part of the program consists of the logic required to perform and terminate the Newton-Raphson iteration. Two criteria must be satisfied to terminate the iteration. The first is

$$\delta m = \left[\sum_{j=1}^{n} \left(\frac{\partial m}{\partial y_j} \delta y_j \right)^2 \right]^{1/2} \leq b m_I \tag{11}$$

where m_I is the mass at the beginning of the variational phases; b (usually 1.0×10^{-4}) is some tolerance factor, which may be specified at execution time; and δm is a measure of the payload error. The term $\partial m / \partial y_j$ is acquired from the set of perturbations. The second criterion is

$$\left(\left| \delta y_r^i \right| \leq Q^i B^i \right) \qquad i = 1, \, n \tag{12}$$

where Q and B are n-vectors. The elements of the Q-vector are parameters that consist of values to which the errors in final conditions are compared. For instance, suppose that radius R was one of the required final conditions y_j. The error δR in R is compared with R; thus $Q^j = R$ and B^j is the tolerance required, usually about 1.0×10^{-4}.

The two convergence criteria ensure that the error in the desired final conditions will be within the prescribed tolerance as well as ensuring that the effect of the accepted error in final conditions on final payload will be within tolerance.

Equation (11) is also used to determine whether the iteration is converging. If the value of δm on an extrapolated trajectory is larger than the δm on a previous trajectory, δx_r is halved successively until the δm for the new trajectory is less than that for the reference. If the extrapolated trajectory does not improve over the reference after a designated number of these operations, the problem is abandoned. The user may determine the number of halving or damping operations allowed.

After the iteration has proceeded to the point that convergence is close, the program may dispense with the perturbations of a reference and just replace the worst trajectory in the set with the new reference, calculate a new matrix M, and proceed in this manner until convergence is obtained.

In addition to the problem of choosing initial guesses, there is the more difficult problem of choosing the perturbation size in order to obtain the perturbed trajectories. The principal source of the difficulty is the strong coupling of the partial derivatives to the optimal phasing equations used to terminate phases in the internal iteration. It is not

unusual for the effect of a particular perturbation size to change quite significantly during convergence because of the great nonlinearity of the optimal phasing effect on the partial derivatives. There are at least two solutions to the difficulty. From a mathematical point of view, the preferable solution would be the integration of analytical partial derivatives. However, this procedure presents very complicated programming problems and significantly increases the complexity of the program. It would also reduce the flexibility of the program in that it would reduce the possibility of the user altering the program significantly. The more practical solution, and the one utilized, was an empirical method of altering the perturbation size in the course of the iteration. In the first set of perturbations, the perturbation size is modified by a simple iteration until

$$5\times10^{-5} \le \frac{\Delta R}{R} \le 5\times10^{-4} \tag{13}$$

where R is the final radius. This is obviously not entirely satisfactory; therefore, on all subsequent interations, the perturbation size is adjusted such that

$$5\times10^{-5} \le \frac{\left[\sum_{j=1}^{n}\left(\frac{\partial m}{\partial y_j} Z_j\right)^2\right]^{1/2}}{m_I} \le 5\times10^{-4} \tag{14}$$

where Z is an n-vector of differences between the final conditions of the reference and a perturbed trajectory, and the $\partial m/\partial y_j$ components are acquired from the previous set of perturbations. Equation (14) is comparable to the equation used to terminate the iteration (eq. (11)), and empirically, provided a satisfactory measure of perturbation size.

Variational functions may be used to terminate some of the optimized phases in order to reduce the iteration size. The functions are monotonic, and the phases are terminated when the functions are equal to zero. The initial conditions affect these functions and determine the values of the functions at the beginning of the phases that must be terminated. If the initial value of the function is positive and the function is monotonically increasing, the equation cannot be satisfied. An analogous situation would exist for a monotonically decreasing function that is initially negative. The first set of initial conditions must be chosen by the user, and the program provides no remedy for a set for which the phasing equations cannot be satisfied. An error message containing the phase number of the offending phase is printed to inform the user that this difficulty has been encountered. This problem most frequently occurs when there is a coast between two stages of a vehicle, or a coast is desired between the burns of a stage, and the user does

not have a set of reasonable initial conditions. The most practical way of avoiding the problem is to fix the duration of a potentially offending phase at a reasonable value, target the problem, then optimize the offending phase, all in one computer run.

If this problem occurs in the course of iteration, the program will reduce the changes in initial conditions until it is possible to satisfy the phasing equation. If a perturbed trajectory encounters the same difficulty, the perturbation size is reduced until the phasing equation can be satisfied.

When the user encounters a new problem in which several phase durations must be optimized and he has little idea of what the initial conditions should be, he may improve the convergence characteristics of the problem by fixing all but one of the phase durations, converge the problem, and free the remainder one by one, converging each time. This is easily accomplished in a single computer submission. Although this procedure would appear to require more time, it may actually save time. Even if the complete problem would have converged without this technique, the number of iterations required to converge the complete problem may exceed the total number required for the parametric procedure. More often, the parametric procedure will be the only way of obtaining solutions in difficult problems because of the need for accurate guesses of initial conditions. This procedure applies only when reasonable estimates of initial conditions are not available.

For a problem that requires a booster table (as described in the BOOSTER section), the kick angle spacing $\Delta\alpha_k$ is a factor in the final accuracy of the solution. The user may check the accuracy of the interpolation scheme by causing the program to integrate the booster with the exact α_k and boost-phase duration specified by the converged initial conditions. At the end of the booster integration, the upper phases are integrated using the converged initial conditions and the exact state variables obtained from the integration. The error checks are performed as before, and, if this trajectory satisfies the tests, the problem is complete. Otherwise, the kick angle spacing $\Delta\alpha_k$ is halved, the convergence procedure is repeated, and the new exact α_k and phase duration are again integrated followed by the upper phases. The error checks are again performed. This process continues until $\Delta\alpha_k$ is small enough to provide the desired accuracy.

FINAL BOUNDARY CONDITIONS

The final boundary conditions for optimizing a multistage launch vehicle are of two types. The most easily understood are those which are dependent on the desired final orbit. More complex are those that consist of equations which must be satisfied in order to ensure optimum phasing. The second type is dependent on the variety of combinations of powered and coast phases, zero and nonzero structure factors, and the order of the

fixed- and optimized-phase durations. These optimum phasing equations are similar and are usually identical to some of those that may be used in the internal optimization of phasing times. Some of these equations may be evaluated before the end of the trajectory, and some of those that are evaluated at the end are dependent on parameters from intermediate points. The program selects the proper equations for evaluation as this is a tedious task. Reference 1 contains the derivations for these equations. Appendix C describes the selection process.

The final orbit is completely specified by any four independent orbit parameters such as R (radius), $\dot{R}$ (radial velocity), ω (angular velocity), and φ (travel angle or polar angle). The variational solution provides means by which any of the orbital parameters left unspecified may be optimized. As shown in reference 1, if the travel angle or polar angle φ is unspecified, $\lambda_4 = 0$ for an optimum φ. Because nonoptimum travel angles are rarely desirable in the type of problem usually considered by this program and, additionally, are difficult to converge, the travel angle φ may not be specified. If necessary, nonoptimum travel angle trajectories may be obtained by loading $\lambda_4 \neq 0$ and observing the change in travel angle.

The following sets of final orbit conditions are included in the program available to the user.

(1) Energy per unit mass E, radius R, and flight path angle Γ. (Travel angle φ is optimized.) This is equivalent to specifying the radius, radial velocity, and angular velocity as shown by the following equations:

$$E = \frac{\dot{R}^2 + \omega^2 R^2}{2} - \frac{\mu}{R}$$

$$\Gamma = \tan^{-1} \frac{\dot{R}}{R\omega}$$

(2) Energy E. (Radius R, flight path angle Γ, and the travel angle φ are optimized).

(3) Energy E and flight path angle Γ. (Radius R and travel angle φ are optimized.)

(4) Energy E and perigee radius R_p. (True anomaly and flight path angle Γ are optimized.) This is equivalent to specifying eccentricity e and semilatus rectum p, but letting true anomaly θ and argument of pericenter $\epsilon = \varphi - \theta$ be unspecified where both φ and θ are to be optimized:

$$p = 2R_p \left(\frac{ER_p}{\mu} + 1 \right)$$

$$e = \frac{2ER_p}{\mu} + 1$$

The argument of pericenter ϵ is defined herein as the angle between the radius vector at the beginning of the variationally steered portion of the trajectory and the pericenter radius. The derivation of the final conditions required to optimize the unspecified orbit parameters is given in appendix C of reference 1.

If the user requires a set of final boundary conditions which is not included, they may be added with very minor revision to the program. The maximum iteration size allowed by the program is five. (This limitation is somewhat arbitrary, but convergence problems increase with the number of iteration variables and five seems to be a practical limit.) A six-phase problem may always be solved with this limitation if it is possible to terminate the final phase on one of the final state conditions (such as energy). The user should keep this in mind in adding any new set of final boundary conditions.

INPUT AND OUTPUT

There are two types of input into the program. Most of the data enter in decimal form by an input routine. The binary booster table is introduced by a special routine which inputs binary cards.

The input routine utilized by this program is described in reference 5. This routine loads fixed- or floating-point numbers, octal numbers, and alphabetic words; it is capable of performing simple arithmetic, as indicated on input cards. Arrays may be loaded quickly in consecutive memory locations. The programming indicated in reference 5 has been made compatible with the IBM 7094 computer, but the capability and instructions for use remain as in reference 5.

The routine for reading binary data and the complementary routine for punching binary cards were developed at Lewis for use on the 7094 computer. They are included with the program. Should these routines prove impossible to adapt to another computer facility, it should be possible to replace these routines with others which perform the same functions. The input routine just described may also be replaced by another of similar capability without great difficulty.

The output from the program also is both binary and decimal. The binary output consists of the information stored in the booster table and is punched out on cards by the routine described previously.

Decimal output is derived from both the booster and upper-phase portions of the code. Most of these data consist of information regarding the state conditions at the beginning, end, and at intermediate points in the integration of the trajectory. Since the

two parts of the trajectory are not integrated consecutively and since the booster is integrated in three dimensions and the upper phases in two, the trajectory output data for the booster and upper phases will be separated and are necessarily different.

A two-line and a five-line output format are available for the booster data. The two-line form prints only those variables of most immediate interest to the user. The five-line output prints most, if not all, of the variables that could interest the user. The list and definitions of the variables printed are given in the comments printed at the beginning of the subroutine OUTPT2.

The trajectory data for the upper phases plus auxiliary data useful in operating the code are printed in a five-line output. The list and definitions of the variables are given in the comments for OUTPT1.

At the beginning of the output for a problem, a synopsis of data for each phase is printed. At the end of each trajectory, a compilation of data of possible interest to the user is printed. These variables and their definitions are given in OUTPT1.

DIRECTIONS FOR UTILIZATION OF DECK

The decimal input data for this program are introduced through subroutine INPUT (described in the section INPUT AND OUTPUT). The required data for utilizing this code may be arbitrarily divided into six categories. In an attempt to provide a foolproof method of loading the required data into the program, the instructions will be treated in six categories and each category will be divided into necessary and optional sections where applicable. Examples are provided where they may be helpful.

The constants in this program are preloaded in U.S. Customary Units. Any other units of length, mass, and force may be used by changing the preloaded constants. This process is explained in the section PROGRAM CONSTANTS. For convenience, the units listed with the input variables described in this section are given in both the U.S. Customary and SI Systems.

DRAG MODEL

Drag is calculated from the equation

$$\text{drag} = C_D q A_{ref}$$

where C_D is the drag coefficient, q is the dynamic pressure, and A_{ref} is the reference area. A curve fit of C_D as function of Mach number is required. This is

provided by dividing the curve into segments of arbitrary length which are approximated by quadratic functions. The method of loading the quadratic coefficients is explained in the description of the variable COEFN.

Necessary Data

Variable	Description
AREA	Reference area to be used in drag calculation, AREA = A_{ref} (e.g., AREA = 78.5), ft^2; m^2
COEFN	COEFN is of the form X_0, a_1, b_1, c_1, X_1, a_2, b_2, c_2, X_2, a_3, b_3, c_3, . . ., a_N, b_N, c_N, X_N. The Mach numbers that designate the segmenting of the drag curve are X_0, X_1, X_2, . . ., X_N. The coefficients in the quadratic are a_i, b_i, and c_i (i = 1, N) where $C_D = a_i + b_i x + c_i x^2$. If the coefficients are not available, they may be generated by the code by loading the optional data.

Optional Data

The coefficients are generated by loading a set of point pairs in ascending order by Mach number from the curve of C_D as a function of Mach number. The following variables must be loaded.

Variable	Description
NSETS	Number of point pairs minus one divided by two. The number of point pairs must be odd.
VARIND	The independent or first members of the point pairs loaded consecutively (Mach number).
VARDEP	The dependent or second member of the point pairs loaded consecutively (drag coefficient).

Examples of coefficient generation: If the following point pairs (1, 2), (2, 1), (3, 2), (4, 4), and (6, 5) are acquired from a curve, the variables would be assigned values as follows: NSETS = 2; VARIND = 1, 2, 3, 4, 6; and VARDEP = 2, 1, 2, 4, 5. If the coef-

ficients are not available and must be generated by the COEFNT routine, the newly generated coefficients are punched out on cards. They are thus available for any subsequent machine submission.

Output Control

The output frequency during a trajectory is controlled by a routine called STEP. Because each trajectory is divided into a booster portion and an upper-phase portion and each is integrated separately, the output frequency may be different for each section. The STEP routine utilizes six internal variables to enable the user considerable flexibility in the frequency and spacing of output during the course of a trajectory. The variables are MODOUT, MODS, STEPS, TMIN, DELMAX, and NOUT. MODOUT may assume values from 1 to 7. These internal variables are set by loading variables mentioned in later sections.

Variable	Description
MODOUT = 1	Output every N^{th} integration step (N = STEPS) until time equals TMIN, then MODOUT is changed to 2
MODOUT = 2	Output every X seconds (X + DELMAX) for duration of that portion of trajectory
MODOUT = 3	Output every X seconds (X = DELMAX) until time equals TMIN, then MODOUT is set equal to 4
MODOUT = 4	Output every N^{th} integration step (N = STEPS) until end of that portion of trajectory
MODOUT = 5	Output at beginning and end of that portion
MODOUT = 6	No output
MODOUT = 7	Allows user to obtain output at intervals of DELMAX beginning at first time which is an integral multiple of the DELMAX interval; useful in upper-phase portion of trajectory, which frequently begins at some nonintegral time

MODS and NOUT (J, I), (J = 1, 6; I = 1, 3) control the type and frequency of output at the phasing points:

Variable	Description
MODS = 1	No output at phasing points
MODS = 2	Output before and after each phasing point
MODS = 3	Output before but not after phasing
MODS = 4	Output after but not before phasing

NOUT is a subscripted variable which allows the user to suppress output at a particular phase. This option is useful when the user may wish to emphasize certain phases by ignoring others, such as a desire to emphasize the staging points and to ignore the trajectory phasing points, which are not vehicle significant.

In order to skip a phasing point J, NOUT (J, I) = 1. NOUT is not an input variable, but is loaded indirectly by loading a similar variable (NPRNTB, NPRNTI, or NPRNTC) which corresponds to an omission desired in either the booster or the upper-phase portion.

BOOSTER DATA

Usually, the booster stage consists of a single segment (i.e., one flow rate and vacuum thrust and no hardware weight to be discarded until the first variational phase). However, the code will accommodate a more complicated booster. The booster in the more complicated configuration may be divided into as many as six segments after the initial vertical rise (which assumes the propulsion characteristics of the first segment), each phase of which may have an independent thrust, flow, and exit area. A fixed weight may be jettisoned at the end of each segment. The necessary data assume a single segment booster.

Necessary Data

Variable	Description
LAT	Latitude of launch site (positive North of equator), preloaded as LAT = 28.310293, deg
LONG	Longitude of launch site (positive East of Greenwich), preloaded as LONG = 279.461759, deg

Variable	Description
AZMUTH	Desired launch azimuth, measured positive clockwise from North (e.g., AZMUTH = 90), deg
WTO	Initial gross weight (e.g., WTO = 6.00 E + 06), lb; kg
EXITS	Total exit area of engines in booster portion, $F = F_V - A_eP$, where F is thrust, F_V is vacuum thrust, A_e = EXITS, and P is atmospheric pressure, ft^2; m^2.
RERUN	Variable determining whether to check kick-angle grid spacing; if RERUN = 1, check booster; if RERUN = 0, no booster check (preloaded)
MODEB(I)	(I = 1, 2) Array which determined output frequency for booster, MODEB(I) = 61 is preloaded.
MODEB(1)	Determines output frequency during vertical rise portion of integration
MODEB(2)	Determines output frequency for printout of converged booster and will be utilized only if RERUN = 1

MODEB(I) is a two digit number: the first digit corresponds to the desired value of the internal variable MODOUT, and the second digit corresponds to the desired value of the internal variable MODS. See the section OUTPUT CONTROL for a discussion of the output options available. As may be noted from this section, certain choices of MODOUT require additional input from the list of optional variables. These variables will be noted in the discussion of the optional data.

Variable	Description
MODEC	Two types of output as well as no output are available for the booster
MODEC = 1	No booster output
MODEC = 2	Preloaded; two-line output option printing those variables which are of immediate interest in generation of a booster table
MODEC = 3	A five-line output option which prints an extensive description of boost trajectory

Variable	Description
PERB(I)	Array which determines beginning and final time points of booster table
PERB(1)	Decimal fraction of initial gross weight which must be expended before table storage is begun; preloaded as 0.5
PERB(2)	Decimal fraction of initial gross weight which must be expended before table storage is ended; preloaded as 0.8

Optional Data

A multisegment booster requires separate data for each segment. These data are supplied in the following class of data.

Variable	Description
LAST	Number of boost phases $1 \leq \text{LAST} \leq 6$; LAST = 1 is preloaded; if LAST > 1, FORCES, FLOWB, EXITB, HARDB, and TBOOST must be loaded
FORCES(I)	(I = 1), LAST - 1) Vacuum thrust levels of multiphase booster, preloaded to zero (e.g., FORCES = 8.6 E + 06, 5.5 E + 06, . . .), lb; N
FLOWB(I)	(I = 1, LAST - 1) Flow rates of a multiphase booster, preloaded to zero (e.g., FLOWB = 4.1 E + 04, 2.68 + 04, . . .), lb/sec; kg/sec
EXITB(I)	(I + 1, LAST - 1) Engine exit areas for a multiphase booster, preloaded to zero (e.g., EXITB = 534.342), ft^2; m^2
HARDB(I)	(I = 1, LAST - 1) Jettison weights of a multiphase booster, preloaded to zero (e.g., HARDB = 100000, 50000, . . .), lb; kg
TBOOST(I)	(I = 1, LAST - 1) Durations of segments multisegment booster (e.g., TBOOST = 60, 80, 20, . . .), sec

The thrust, flow rate, and hardware weight of the last segment of the booster (data for the table are stored during this phase) are loaded with the upper-phase data.

Variable	Description
TKTIME	Time duration of vertical rise, TKTIME = 15 is preloaded, sec
ASTART	Altitude at beginning of vertical rise, ASTART = 0 is preloaded, ft; m
TSTART	Time at beginning of vertical rise, TSTART = 0 is preloaded, sec
VSTART	Relative velocity at beginning of vertical rise, VSTART = 0 is preloaded, ft/sec; m/sec

The remaining variables supply values to the internal variables, DELMAX, TMIN, and STEPS at the appropriate time.

Variable	Description
DELMXB(I)	(I = 1, 2) DELMXB supplies appropriate time intervals to DELMAX. (See section on OUTPUT CONTROL.) Subscripts of DELMXB refer to rise time and converged trajectory as in the MODEB discussion. DELMXB = 10 is preloaded, sec
TMINB(I)	(I = 1, 2) TMINB supplies appropriate time to TMIN. (See section on OUTPUT CONTROL.) Subscripts of TMINB refer to rise time and converged trajectory as in MODEB discussion, sec
STEPB(I)	(I = 1, 2) STEPB supplies appropriate value to STEPS. (See section on OUTPUT CONTROL.) Subscripts of STEPB refer to rise time and converged trajectory as in MODEB discussion.

UPPER-PHASE DATA

Necessary Data

Variable	Description
TB(I)	(I = 1, 6) Durations for each phase, sec

Variable	Description
TB(1)	Refers to last booster phase. If TB(K + 1) = 0, total number of phases = K, $2 \leq K \leq 6$ (e.g., TB = 100, 150, 30). Note that duration for each phase must be loaded whether duration is fixed or optimized. If a phase is optimized, duration serves as guess required for iteration procedure.

THRUST, FLOW, PROP, and NOPT are subscripted variables, the first member of which refers to the last booster segment. The total loaded in a given problem equals K, the number of phase durations.

Variable	Description
THRUST(I)	(I = 1, K) Vacuum thrust levels, preloaded to zero, lb; N
FLOW(I)	(I = 1, K) Flow rates, preloaded to zero, lb/sec; kg/sec
PROP(I)	(I = 1, K) Hardware structure factors, where PROP(I) equals k in equation $$W_s = W_H + kW_p$$ where W_s is total structure weight of a phase, W_H is a fixed weight, k is propellant sensitive structure factor, and W_p is propellant weight for phase. PROP is preloaded to zero.
HARD(I)	(I = 1, K) Fixed hardware weight where HARD(I) equals W_H in preceding equation, HARD(I) is preloaded to zero, lb; kg
NOPT(I)	(I = 1, K) NOPT(I) = 1 indicates that I^{th} phase duration is to be optimized. NOPT(I) = 0 indicates that I^{th} phase duration is fixed.
PSI	Guess at initial thrust angle, measured upwards from local horizontal, (e.g., PSI = 0.525), rad
PSID	Guess at time rate of change of thrust angle, (e.g., PSID = 0.0014.), rad/sec
TKICK	Guess at kick angle of booster, (e.g., TKICK = 88.6), deg

Variable	Description
ITERPD	If fixed kick angle is desired, ITERPD = 1 must be loaded. If kick angle is to be optimized, ITERPD = 0, which is preloaded.
MODE(I)	(I = 1, 3) Variable which determines output frequency for upper-phase portion of trajectory.
MODE(1)	Determines output frequency for printout during process of iteration. MODE(1) = 61 is preloaded.
MODE(2)	Determines output frequency for converged upper-phase printout. MODE(2) = 51 is preloaded.
MODE(3)	Determines output frequency for converged trajectory which exceeds one of propellant constraints imposed by loading WPMAX. MODE(3) = 61 is preloaded.

MODE is a two-digit number performing for the upper phases the function MODEB does for the booster. (See MODEB and the section on OUTPUT CONTROL .)

Optional Data

Variable	Description
DROP	Weight dropped between integrated portion of trajectory and impulse velocity increment, DROP = 0 is preloaded, lb; kg
WPMAX(I)	(I = 1, K) An upper bound may be placed on propellant loading of any optimized phase. If that is exceeded, duration of that phase will be fixed at time which will exactly consume specified maximum allowed propellant and remaining phases open to optimization will be reoptimized. WPMAX(I) = 0 is equivalent to not specifying maximum propellant for I^{th} phase, and zero is preloaded, lb; kg

If the last phase duration is optimized and the maximum propellant loading is specified, it is important to note that this maximum propellant loading refers only to the propellant required to satisfy the specified final conditions and does not include any of the propellant required for any additional impulsive velocity increment. If the propellant

maximum is exceeded, the last phase duration will be fixed so that the specified propellant will be used before the impulsive velocity increment is added.

DELMX, TMIN, and STEP supply values to the internal variables DELMAX, TMIN, and STEPS, respectively, at the appropriate times.

Variable	Description
DELMX(I)	(I = 1, 3) DELMX supplies appropriate time interval to DELMAX. (See section on OUTPUT CONTROL.) Subscripts of DELMX refer to unconverged, converged, and WPMAX exceeded printout as in MODE discussion, sec
TMIN(I)	(I = 1, 3) TMIN supplies appropriate time to internal TMIN. (See section on OUTPUT CONTROL.) Note that loaded TMIN is subscripted and does not have same location as TMIN in STEP. Subscripts of TMIN refer to unconverged, converged, and WPMAX exceeded printout as in MODE discussion, sec
STEP(I)	(I = 1, 3) STEP supplies the appropriate value to STEPS. (See section on OUTPUT CONTROL.) Subscripts of STEP refer to unconverged, converged, and WPMAX exceeded printout as in MODE discussion.

FINAL CONDITIONS

There are four sets of final conditions available in the program. These are listed and explained in the section FINAL BOUNDARY CONDITIONS.

Necessary Data

Variable	Description
NFINAL	Determines which of final conditions will be used.
NFINAL = 1	Preloaded; maximum payload to specified energy, radius, and flight path angle; equivalent to specified radius, radial velocity, and angular velocity. For convenience, program may be loaded either way.

Variable	Description
FYD(I)	(I = 1, 3) If radius, radial velocity, and angular velocity are desired final conditions, they are loaded as follows:
FYD(1)	Radius, ft; m
FYD(2)	Radial velocity (preloaded to zero), ft/sec; m/sec
FYD(3)	Angular velocity (preloaded to zero), rad/sec

It is very unusual to desire a final condition in which the angular velocity is equal to zero. FYD(3) = 0 is therefore used as a trigger to inform the program that the user wishes to specify energy, radius, and flight path angle in which case the following are loaded:

Variable	Description
ENERGY	Energy, ft^2/sec^2; m^2/sec^2
FYD(1)	Radius, ft; m
ANG	Flight path angle, preloaded to zero, rad
NFINAL = 2	Maximum payload to a specified energy; energy must be specified
ENERGY	Energy, ft^2/sec^2; m^2/sec^2
NFINAL = 3	Maximum payload to some specified energy and flight path angle; energy and flight path angle are loaded
ENERGY	Energy, ft^2/sec^2; m^2/sec^2
ANG	Flight path angle, rad
NFINAL = 4	Maximum payload to some specified energy and perigee radius; energy and perigee radius are loaded
ENERGY	Energy, ft^2/sec^2; m^2/sec^2
FYD(1)	Perigee radius, ft; m

Optional Data

Variable	Description
DELTAV	After final conditions specified by selecting NFINAL are reached, an additional impulsive velocity increment may be added with final integrated phase. If DELTAV is loaded, propellant required for velocity increment will be considered in optimization of last phase propellant loading. DELTAV = 0 is preloaded, ft/sec; m/sec
RESERV	Identical to DELTAV except that additional propellant required in last phase for velocity increment is not included in phase propellant optimization. RESERV = 0 is preloaded, ft/sec; m/sec

PROGRAM CONTROL VARIABLES

Variable	Description
IMODE	Used to control loading and generation of binary booster tables (see section BINARY BOOSTER TABLE)
IMODE = 3	Loaded when a binary booster table is to be loaded
IMODE = 2	Never loaded
IMODE = 1	Never loaded
IMODE = 0	Loaded when new booster is desired in course of machine submission and no table is provided for new booster. Note that IMODE is not loaded when there is no table available for first problem in a submission and is not loaded as long as same booster was used in problem preceding one under consideration.
NDUMP	Controls production of binary cards
NDUMP = 1	Binary cards punched, preloaded
NDUMP = 0	No binary cards punched
DELTB	Initial integration step size for booster, DELTB = 2.0 is preloaded, sec

Variable	Description
DELTST	Initial integration step size for upper phase, DELTST = 10.0 is preloaded, sec
DELTK	Grid spacing for booster table kick angles (see section BOOSTER)DELTK = $0.1/2^i$, i = 0, 1, 2, . . . , DELTK = 0.1 preloaded, deg
DX(I)	(I = 1, 5) Supplies initial perturbation sizes for finite-difference scheme $X_p = X_n + DX(J) \cdot X_n$, where X_n is reference value of initial condition and X_p is perturbed value. (DX(I) = 1.0 E-05, I = 1, 5) is preloaded.
CLEAR	If at beginning of new problem, user wishes to restore initial values to DX array, load CLEAR = 1. If CLEAR = 1 is not loaded, final values of DX from previous problem will be used, as is frequently desirable.
CMAX	Each time new reference trajectory is integrated in course of solving boundary value problem, internal variable COUNT is incremented by 1. If convergence is not obtained when COUNT equals CMAX, problem is abandoned and program reads in next set of input data. CMAX = 20 is preloaded.

NDAMP, ERR, and ERROR are variables which rely either directly or indirectly on equation (11) from the section ITERATION PROCEDURE.

$$\delta m = \left[\sum_{j=1}^{N} \left(\frac{\partial m}{\partial y_j} \delta y_j\right)^2\right]^{1/2} \leq bm_I$$

This equation is used to determine whether the iteration is converging, whether it remains advantageous to perturb the reference trajectories, and also as one of the two criteria for admitting convergence and checking the booster table grid spacing.

Variable	Description
NDAMP	In ITERATION PROCEDURE, a procedure described as damping is discussed in which changes in initial conditions from one reference trajectory to the next are halved if new trajectory is not an improvement over old. Number of times this operation may be repeated before problem is abandoned is limited by NDAMP. NDAMP = 5 is preloaded.
ERR	Perturbations of reference trajectories may be discontinued when $\delta m \leq ERR * m_I$, where ERR is preloaded equal to 1.0 E-04.
ERROR(I)	(I = 1, 2) Constitutes variable factor in first convergence test described in ITERATION PROCEDURE. Test is satisfied if $\delta m < ERROR(1) * m_I$
ERROR(1)	1.0 E-04 is preloaded.
ERROR(2)	Constitutes variable factor in second part of check on acceptability of grid spacing of kick angles in booster table. The spacing is accepted if $\delta m < ERROR(2) * m_I$. ERROR(2) = 1.0 E-03 is preloaded.
TOL(I, J)	(I = 1, 5; J = 1, 2) Second convergence test described in ITERATION PROCEDURE section (eq. (12)) requires that $(\lvert \delta y_r^i \rvert \leq Q^i B^i)$ where i = 1, n. This test is used for both convergence of Newton-Raphson scheme and as a check on grid spacing of booster table. TOL is equivalent to B-vector.
TOL(I, 1)	Constitutes variable factor in second convergence test. Test is satisfied if $y_r^i \leq TOL(i, 1) * Q^i$ where i = 1, N. TOL(I, 1) = 1.0 E-04, I = 1, 5 is preloaded.
TOL(I, 2)	Constitutes variable factor in first part of check on acceptability of grid spacing of kick angles in booster table. Spacing satisfies test if $y_r^i \leq TOL(i, 2) * Q^i$ where i = 1, N. TOL(I, 2) = 1.0 E-03, I = 1, 5 is preloaded.
NPRINT	User has option of seeing perturbed trajectories. NPRINT = 1 causes the program to print these trajectories if the unconverged trajectories are printed. NPRINT = 0 (preloaded) omits the perturbed trajectories.

Variable	Description
OBLATN	If OBLATN = 1 (preloaded), the oblate earth model given in reference 3 is used. A spherical model is used if OBLATN = 0.
NPRNTB(I)	(I = 1, 6) NPRNTB supplies NOUT values for boost trajectory (for a multiphase booster). (See section on OUTPUT CONTROL.) If user wishes to print some phasing points but wishes to omit others, he may load NPRNTB(I) = 1 to omit phasing printout on Ith booster phase. NPRNTB(I) = 0, I = 1, 6 is preloaded.
NPRNTI(I)	(I = 1, 6) NPRNTI supplies NOUT values for unconverged upper-phase trajectories. (See section on OUTPUT CONTROL.) It is similar to NPRNTB and is similarly preloaded.
NPRNTC(I)	(I = 1, 6) Serves same purpose as NPRNTI except that it controls printout for converged upper phases. NPRNTC(I) = 0, I = 1, 6 is preloaded.

EREF, ERLIMT, and STEPMX are the control variables for the integration scheme. Appendix D of reference 3 contains a description of the integration method. An adequate understanding of these variables would require a knowledge of the information in appendix D of reference 3. Therefore, only the preloaded values of these variables are listed here:

Variable	Description
EREF	EREF = 1.0×10^{-5} is preloaded.
ERLIMT	ERLIMT = 3×10^{-5} is preloaded.
STEPMX	STEPMX = 200 is preloaded.
ERRMXK	Internal optimization requires that phases be terminated when designated parameters reach certain values. This is accomplished in a routine called COAST. The tolerance within which these must agree is equal to ERRMXK, which is preloaded equal to 1.0 E-05.

PROGRAM CONSTANTS

The program requires a few constants which are all preloaded, but which may be altered at execution.

As discussed at the beginning of the directions, the program is preloaded in U.S. Customary units. SI units may be used. In the U.S. system, weight and force are loaded in the same units. In the SI system, force may be loaded in newtons or in kilograms force where there is a factor of g between the two when they represent the same force. In order to accommodate either convention, if the weight (or mass) is loaded in the same units as the force, then G must be loaded equal to the equivalent of 32.174. If force and weight (or mass) are loaded in different units, G should be set equal to 1. This rule would, of course, apply to any set of units. CONM and CONN are preloaded for the U.S. system of units.

Variable	Description	Preloaded value	Units
FM	Gravitational constant for Earth	1.4076539E+16	ft^3/sec^2
RO	Earth radius used in upper-phase portion of deck	20.90989E+06	ft
A	Semimajor axis of oblate earth model	20.925738E+06	ft
B	Semiminor axis of oblate earth model	20.855568E+06	ft
G	Acceleration of gravity at Earth's surface	32.174	ft/sec^2
REVOLV	Angular velocity for Earth	7.29211512E-05	rad/sec
ROA	Reference radius for use in interpolation scheme	20.969890×10^6	ft
CONM	Conversion from meters to unit of length for problem	3.2808398	ft/m
CONN	Conversion from newtons to unit of force for problem	0.22481905	lb/N

BINARY BOOSTER TABLE

After a booster table has been generated, the user has the option of punching the table out on binary cards which may be read back into the program if the same booster should be reconsidered. (See IMODE and NDUMP in the section PROGRAM CONTROL

VARIABLES.) The subroutines which read and punch binary cards vary with the computer installation and therefore are not described herein. Copies of the Lewis Research Center routines will be provided with copies of the code, and the user must modify or replace them with routines of similar capability.

CODING

Appendixes D to H contain aides to utilizing the code. A listing of the program is provided in appendix H. All the required FORTRAN IV subroutines are arranged in alphabetic order. BCDUMP, BCREAD, and INPUT are not in FORTRAN and are not listed. Figure 1 is a block diagram showing the major routines in the program. The

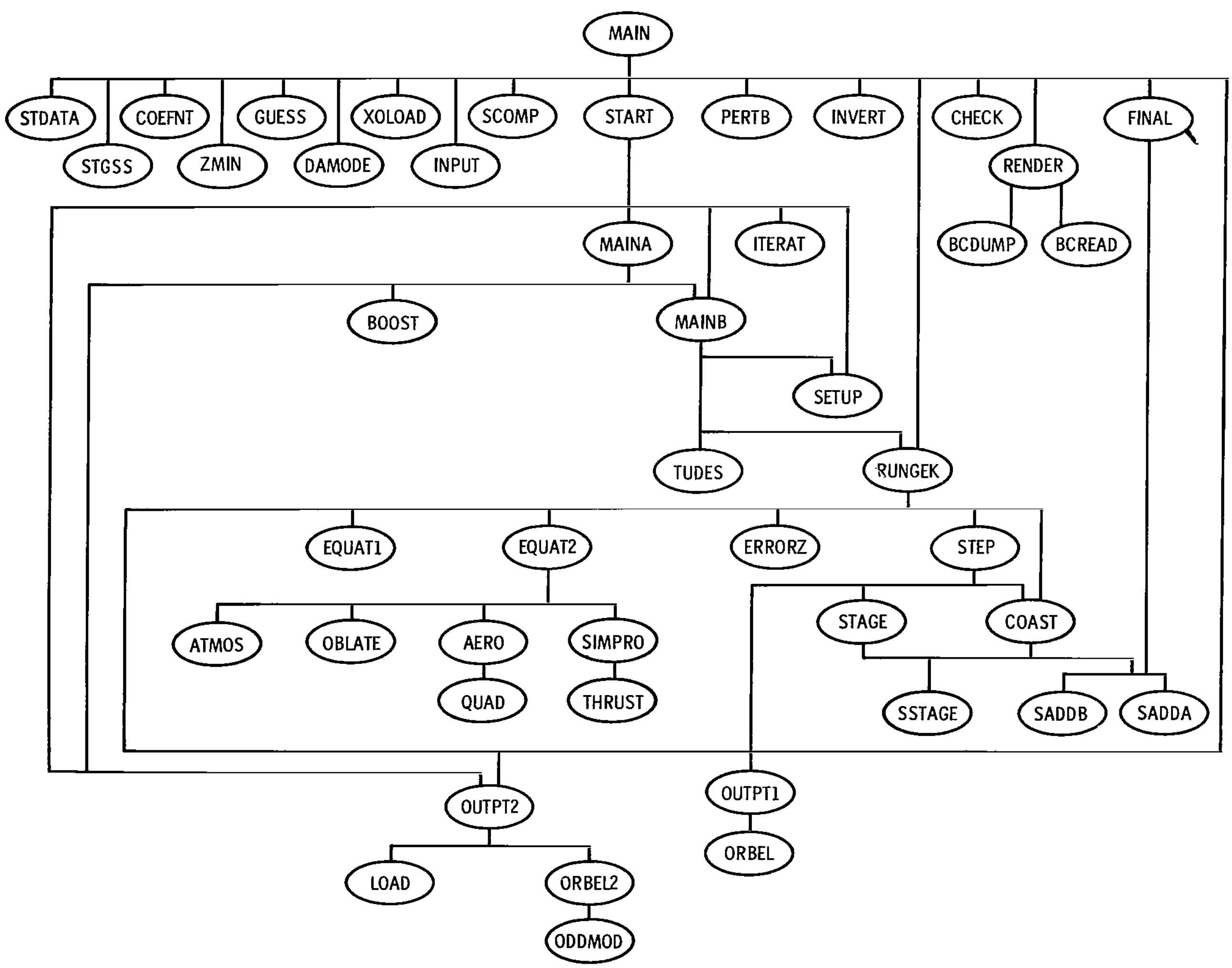

Figure 1. - Program block diagram.

INPUT routine requires an input table which is placed just before the data, as described in reference 5. A listing of this table is provided in appendix F.

There are three main common blocks in the code: RUNG, ATABLE, and CSTAR. RUNG contains variables which are used in the integration and step size control. ATABLE contains those variables which may be stored on cards at the end of a problem and read into the computer later. The remaining program variables requiring common locations are in the CSTAR block. This block common is available to loading from the INPUT routine.

Two lists of the variables in the three common blocks are provided. A glossary of variables in alphabetic order is provided in appendix D. A list of all the equivalence entries in the program in numerical order is provided in appendix E giving the name, subblock, location, dimension, and subroutine for each entry.

Variable names are sometimes duplicated and the same storage may be occupied by unrelated variables or by the same variable appearing under another name. These duplications result from the fact that the code is a combination of two codes and from the difference in coordinate systems. The variables being integrated change depending on whether a booster or upper phase is being integrated. The two lists of variables provided should clarify most, if not all, of the ambiguities introduced by the variable name and location duplication.

SAMPLE PROBLEM

A sample problem has been provided to enable the user to observe the way in which data are loaded into the program and some of the conveniences of the input routine. The listing for the sample problem is given in appendix G. Examples of addition, multiplication, and division are provided in the input data.

The sample problem consists of a three-stage booster to Earth escape passing through a 121-nautical-mile (224 092-m) circular parking orbit. The maneuver from orbit to escape is simulated by introducing an impulsive velocity increment at the end of the last integrated phase. The vehicle consists of a hypothetical kerosene - liquid-oxygen first stage, a hydrogen-oxygen second stage, and a nuclear third stage.

The data are listed in the order in which they are introduced in the set of directions. All the necessary data are explicitly introduced, although some data might have been skipped by utilizing the preloaded values. A few of the optional data entries are utilized. The input data and vehicle parameters should be easily understood if an item by item comparison is made between the data and the DIRECTIONS FOR UTILIZATION OF DECK.

The first example illustrates the case where all the stages are optimized. The booster output is an example of the abbreviated output. The initial kick angle spacing is

also too large causing the program to cut the spacing and run another booster. The second case fixes the first-stage duration and optimizes the last two. The third case fixes the duration of the first two stages and optimizes the third. The full booster output is used for the last two cases.

Lewis Research Center,
National Aeronautics and Space Administration,
Cleveland, Ohio, May 1, 1968,
180-31-01-01-22.

APPENDIX A

SYMBOLS

A_e engine exit area, ft^2; m^2

A_{ref} vehicle reference area, ft^2; m^2

a coefficient in interpolation equation (eq. (4))

B convergence tolerance

b convergence tolerance

C_D total drag coefficient

E energy, ft^2/sec^2; m^2/sec^2

e eccentricity

F thrust, lb force; N

k propellant sensitive structure factor

M matrix relating changes in final and initial conditions

m mass, slugs; kg

n iteration size

P atmospheric pressure, lb/ft^2; N/m^2

p semilatus rectum, ft; m

Q comparison parameters for convergence determination

q dynamic pressure, lb/ft^2; N/m^2

R radius, ft; m

R_p perigee radius, ft; m

t_B booster phase duration, sec

W_H fixed structure weight for a phase, lb; kg

W_p propellant weight for a phase, lb; kg

W_s total structure weight for a phase, lb; kg

x problem variable

y problem variable

Z n-vector of differences

α_k booster kick angle, deg

Γ flight path angle, rad

ϵ argument of perigee, rad

θ true anomaly, rad

λ Lagrange multiplier

μ Earth force constant, ft^3/sec^2; m^3/sec^2

τ burning time, sec

φ polar angle, rad

ψ thrust direction, rad

ω angular velocity, rad/sec

Subscripts:

I initial

r difference between reference and desired values

v vacuum

Superscripts:

i variable number

j variable number

$(\dot{\ })$ derivative with respect to time

APPENDIX B

BOOSTER STEERING

As described in the section on the booster, the boost trajectory begins with a short vertical rise of fixed time after which the vehicle is tilted or kicked over instantaneously at some small predetermined angle in the desired azimuth direction. During the vertical rise, the unit thrust vector $\hat{f} = \hat{R}_o$, where $\hat{R}_o$ is a unit vector in the direction of the radius vector at launch. At the end of the vertical rise, the relative velocity vector is rotated to the desired kick angle (measured from the horizontal) in the desired azimuth direction.

For the remainder of the boost trajectory, the unit thrust vector $\hat{f}$ is constrained to the launch azimuth plane. Hence,

$$\hat{f} \cdot \hat{N} = 0 \tag{B1}$$

where $\hat{N}$ is a unit vector normal to the launch azimuth plane. The component of the thrust vector that lies in the $R - V_R$ (pitch) plane lies along the relative velocity vector; that is,

$$\hat{f} = K_1\hat{V}_R + K_2\hat{h} \tag{B2}$$

where V_R is a unit vector in the direction of the relative velocity and h is the unit relative angular momentum vector given by

$$\hat{h} = \frac{\vec{R} \times \vec{V}_R}{|\bar{R} \times \bar{V}_R|} \tag{B3}$$

The coefficients K_1 and K_2 are evaluated by using equations (B1) and (B2). Evaluating K_1 and K_2 and substituting them into equation (B2) yields

$$\hat{f} = \frac{(\hat{h} \cdot \hat{N})\hat{V}_R + (\hat{V}_R \cdot \hat{N})\hat{h}}{\sqrt{(\hat{h} \cdot \hat{N})^2 + (\hat{V}_R \cdot \hat{N})^2}} \tag{B4}$$

The equations for $\hat{N}$ in an equatorial rectangular coordinate system where the X-Z plane passes through Greenwich are

$$N_x = \sin \Lambda \cos \beta - \cos \Lambda \sin L \sin \beta$$

$$N_y = -\sin \Lambda \sin L \sin \beta - \cos \Lambda \cos \beta \tag{B5}$$

$$N_Z = \cos L \sin \beta$$

where L and Λ are launch latitude and longitude, respectively, and β is launch azimuth angle measured from North in the horizontal plane.

APPENDIX C

SELECTION OF OPTIMIZING EQUATIONS

As described in the section UPPER PHASES, the Newton-Raphson iteration size is decreased by resorting to internal optimization procedures. These are of two types. The simplest to implement are those which require either satisfaction of a closed-form equation or a very simple iteration at the beginning of a trajectory. These are adequately described in appendix B of reference 1. Of more complexity, both in selection and implementation, are those which require that phases of the trajectory be terminated by optimizing equations. These are derived as explained in the section Boundary Value Problem of reference 1. It will be assumed that the reader has both access to and some knowledge of that report. The purpose of this appendix is to demonstrate how the optimization equations (eqs. (45) to (51)) found in the analysis report are implemented.

The optimization equations are evaluated at the beginning or end of the particular phases with which they are associated. The first optimized phase (a phase of unspecified duration) does not have an accompanying equation that may be evaluated during that phase. The remaining optimized phases all supply an optimizing equation that must be satisfied for optimum payload. If the structure factor for an optimized phase is nonzero, its optimizing equation is evaluated at the end of the phase; if it is zero, the equations must be evaluated at the beginning of the phase. As explained in reference 1 and noted herein, optimized phases may be terminated by evaluating the optimizing equations during the appropriate phases and terminating the phase when the equation reaches the desired value. As also noted earlier, phases that may be terminated in this manner remove an initial condition and a final condition from the Newton-Raphson scheme. Of course, those optimizing equations that may not be satisfied by terminating a phase must be satisfied in the Newton-Raphson iteration.

Two examples are provided to demonstrate how the choices are made as to which equations are used to terminate phases and which phases are placed in the Newton-Raphson iteration. These examples are for six-phase problems. Problems that require fewer than six phases are treated in an identical manner.

The first example assumes that the kick angle is fixed.

Phase	1	2	3	4	5	6
Optimized (O), fixed (F)	0	0	0	0	0	0
Structure factor, k	0	0	$\neq 0$	0	0	$\neq 0$
Evaluated beginning (B), end (E)	-	B	E	B	B	E
Equation number from reference 1		(51)	(47)	(48a)	(48a)	(50)

From this chart, it may be seen that equation (51) must be satisfied between the booster and the first upper phase. This equation is satisfied by choosing the initial thrust angle ψ, such that the equation is satisfied, rather than by varying the booster duration as might be done. The booster duration is part of the Newton-Raphson iteration.

There is no equation between the second and third phases; therefore, the second phase duration must also be in the Newton-Raphson iteration. There are two available equations between the third and fourth phases. Equation (47) may be used to terminate phase three, and equation (48a) must be satisfied in the iteration, where equation (48a) refers to phase four. Equation (48a) (referenced to phase five) must be satisfied between phases four and five and hence phase four may be terminated by using this equation. There is no equation between phases five and six; therefore, the duration of phase five must be part of the iteration.

The sixth phase may be terminated by using equation (50) or it may be terminated when the desired energy is acquired. Either of these considerations may be used successfully to terminate the phase, while the other must be placed in the iteration. Termination on energy proved to be more satisfactory from a convergence standpoint. It is used to terminate, leaving equation (50) to the iteration.

In the second example, the kick angle is optimized.

Phase	1	2	3	4	5	6
Optimized (O), fixed (F)	F	F	0	0	0	F
Structure factor, k	$\neq 0$	0	$\neq 0$	0	$\neq 0$	$\neq 0$
Evaluated beginning (B), end (E)				B	E	
Equation number from reference 1				(48a)	(47)	

The fixed phases provide no optimization equations. The first and second phases are of fixed duration and therefore are terminated at the specified times. Note that, although phase three is of optimized duration, it is the first optimized phase and therefore provides no optimization equation. Equation (48a) (referenced to phase four) may be used to terminate phase three since it is evaluated at the beginning of phase four. There is no equation to be evaluated between phases four and five. The duration of phase four must be placed in the iteration. Between phases five and six there is an equation available for terminating phase five. The sixth phase is of fixed duration and thus is terminated on time.

Suppose that there is an equation that must be satisfied between a fixed phase and an optimized phase. Obviously, the equation must be satisfied in the iteration since the fixed phase would most conveniently be terminated on time.

APPENDIX D

GLOSSARY OF VARIABLES

/RUNG/

Common Block

Variable	Common location	Definition
A1	101	Error control parameter in integration scheme (ref. 3)
A2	102	Error control parameter in integration scheme (ref. 3)
DEL1	103	Time interval to next DELMAX printout
DELSTO	104	Correction factor for DEL1
E2	105	Largest of relative errors between Runge-Kutta and a second-order integration (ref. 3)
H2	106	Value of DELT for previous step
I	107	Control parameter to restart integration scheme at discontinuities and to override step size control
NSTAG1	115	Value of NSTAGE for previous step
NSTEP1	108	Control parameter for output control
NSTEP2	109	Control parameter for output control
NSTEP3	110	Control parameter for output control
RATIO	111	Ratio of adjacent step sizes (ref. 3)
RELERR(100)	1	Relative error array, array of quotients of absolute differences between Runge-Kutta and second-order integration and some comparison parameter
SCRIBE	114	Control parameter for printout at beginning of integration and when scheme must be restarted
STEPGO	112	Total number of good integration steps
STEPNO	113	Total number of bad integration steps

Variable	Common location	Definition
XINC(100)	1	Increments of integration variables per step (in RUNGEK), before ERRORZ is called; absolute differences between Runge-Kutta and second-order integration are placed in this storage

/ATABLE/

Common Block

Variable	Common location	Definition
ALT	5	Altitude at end of vertical rise
ANGLEB(4)	25	Storage for latitude, longitude, azimuth at end of vertical rise (similar to ANGLES)
APR	4	Storage for semimajor axis for ellipsoidal planet model or for radius of spherical planet model
APT(3, 100)	201	APT(1, J) is storage for kick angles (note that this variable occupies same storage as KPT)
DELTB	16	Time interval between entries in table
DELTK	199	$\Delta\alpha_k$ for table in storage
FIRST(28)	1	General array of conditions at end of vertical rise
FOURTH(7500)	501	General array for state conditions stored in table
FUELT	14	Propellant used prior to last phase (in which data in table are stored)
HARDBT	15	Total hardware dropped before table storage begins
HSTORE	12	Heat integral to end of vertical rise
IKICK	200	Number of kick angles in table

Variable	Common location	Definition
KPT(3, 100)	201	KPT(2, J) index noting position of data in storage, KPT(3, J) indicator for acceptability of data (indicates boost trajectory may have been unable to complete specified time)
NKICK	20	Number of kick angles which may be stored in table
NKICKS	21	One of dimensions of VAR (see BOOST)
OBLATN	2	Control parameter for oblateness
RADIUS	13	Radius at end of vertical rise
SECOND(2)	199	General array for DELTK and IKICK
TBLAST	18	Time for end of table storage
TBO	17	Weight at beginning of table storage
THIRD(300)	201	General array for APT and KPT
TMINST	19	Storage for TMIN
TSPM	10	Time at beginning of last booster phase (table storage phase)
UN(3)	22	Unit vector perpendicular to launch azimuth plane
VAR	501	Storage for state conditions stored in table
VEL	1	Velocity at end of vertical rise
WSTORE	11	Weight at end of vertical rise
XSTORO(4)	6	Radius vector (X-, Y-, and Z-components) and magnitude at launch

/CSTAR/

Common Block

Variable	Common location	Definition
A	813	Semimajor axis of ellipsoidal planet
ALT	805	Altitude above oblate planet

Variable	Common location	Definition
ANGLES(4)	786	Angles (1) latitude, Angles (2) longitude, Angles (3) azimuth, and Angles (4) elevation
APTMAX(3)	1126	List of kick angles used in interpolation
AREA	809	Drag reference area
ASTART	798	Altitude at lift-off
AZIMI	921	Inertial azimuth heading (see ORBEL2)
AZIMR	922	Relative azimuth heading (see ORBEL2)
B	814	Semiminor axis of ellipsoidal planet model
BETA	880	Desired final flight path angle
BETAI	917	Inertial flight path angle, three-dimensional model
BETAR	918	Relative flight path angle, three-dimensional model
CAPPA	891	General staging parameter (see COAST)
CD	812	Drag coefficient
CLEAR	930	Control parameter to reload DX array (see PROGRAM CONTROL VARIABLES
CMAX	928	Total allowed iterations of Newton-Raphson scheme
COEFN(500)	1501	Coefficient array for CD calculation (see QUAD)
CONM	717	Conversion from meters to unit of length for problem
CONN	718	Conversion from newtons to unit of force for problem
COMP(5)	1031	Comparison parameters for convergence test
COMPA(3)	783	Total nongravitational acceleration in X-, Y-, and Z-directions, three dimensional
CONST(5, 2)	894	Constant defined in reference 1 (see CONEVL) equations (11) and (41c)
CPA	719	Factor for converting pressure to required units from kilograms force per square meter
CPSI	888	Cosine ψ
DELMAX	702	See section OUTPUT CONTROL
DELMX(3)	1163	See section UPPER-PHASE DATA

Variable	Common location	Definition
DELMXB(2)	1174	See section BOOSTER DATA
DELT	701	Integration step size
DELTAV	861	See section FINAL CONDITIONS
DELTBT	803	Initial integration step size, booster
DELTKT	1058	See DELTK in section PROGRAM CONTROL VARIABLES
DELTST	931	Initial integration step size, upper phase
DOMEGA	505	$\dot{w}$ where w = OMEGA
DPHI	503	$\dot{\varphi}$ where φ = PHI
DR	502	$\dot{R}$
DRAG(3)	777	Accelerations due to drag, X-, Y-, and Z-components
DRMASS	501	$\dot{W}$, where W = weight
DROP	863	See section UPPER-PHASE DATA
DTIME	509	Derivative of time, equal to 1
DU	504	$\ddot{R}$
DX(5)	938	Perturbation parameter, see directions in section PROGRAM CONTROL VARIABLES
DZLAM1	506	$\dot{\lambda}_1$ (ref. 1)
DZLAM2	507	$\dot{\lambda}_2$ (ref. 1)
DZLAM3	508	$\dot{\lambda}_3$ (ref. 1)
E	905	Eccentricity
ELEV	790	Kick angle, internal variable
ENERGY	892	See section FINAL CONDITIONS
EREF	713	Reference error parameter (ref. 3)
ERLIMT	706	Max integration error parameter (ref. 3)
ERLOG	707	Natural logarithm of EREF
ERR	1041	Iteration control parameter (see section PROGRAM CONTROL VARIABLES)
ERRMXK	1054	Tolerance on parameters which terminate phases internally

Variable	Common location	Definition
ERROR(2)	1042	ERROR(1) convergence tolerance, ERROR(2) tolerance for checking kick angle spacing
ESTART	795	Initial elevation angle
EXITS(6)	727	Engine exit areas (see section BOOSTER DATA)
FDM	886	Acceleration during upper phases
FIXDTK	1071	FIXDTK = 1, α_k and TB(1) fixed so only one booster needs be run, FIXDTK = 2, no more boosters will be integrated, and FIXDTK = 3, normal mode
FLOMX(6)	837	Flow rates for upper phases (see section UPPER-PHASE DATA)
FLOW	752	Instantaneous flow rate during booster phase
FLOW	877	Instantaneous flow rate during upper phases
FM	715	Gravitational constant
FORCE(3)	774	Accelerations due to thrust in X-, Y-, and Z-directions
FORCES(6)	739	Thrusts during booster phase (see section BOOSTER DATA)
FUEL(6)	871	Weights of propellants expended during upper phases
FUELDV	878	Propellant required for impulsive maneuver
FY(6, 6)	943	Array containing deviations from desired final conditions for reference and perturbed trajectories (used in Newton-Raphson scheme)
FYD(5)	1036	Array containing desired final conditions
G	716	Nominal acceleration of gravity at surface of Earth
H(5)	769	Angular momentum vector H(1) X-component, H(2) Y-component, H(3) Z-component, $H(4) = H(1)^2 + H(2)^2 + H(3)^2$, and $H(5) = \sqrt{H(4)}$
HARD(6)	843	Hardware weights for upper phases
HARDB(6)	721	Hardware weights for multiphase booster model
HEAT	401	Instantaneous value of heat integral during booster
HORV	1156	Horizontal velocity stored in table

Variable	Common location	Definition
IBURN	1072	Counter for loading table
ICC(20)	1201	ICC(1) location in COEFN of coefficients used in CD interpolation (see subroutine COEFNT)
IDATA(6, 5)	1086	Array used to determine which phases are placed in iteration and which are terminated internally (see appendix C)
IKCKST	1066	Temporary storage for IKICK, number of kick angles for which trajectories have been integrated
IMODE	1061	IMODE = 0, integrating vertical rise; IMODE = 1, integrating booster; IMODE = 2, integrating upper phases; and IMODE = 3, table to be loaded
ITB(3)	1123	Counter to indicate discrete burning times to be used in interpolation in BOOST
ITK(3)	1120	Counter used to determine kick angles used in interpolation in BOOST
ITER	1068	Number of phase durations in Newton-Raphson iteration
ITERAD	1067	ITERAD = 1, kick angle in iteration, booster duration optimized in START; ITERAD = 2, kick angle α_k in iteration; ITERAD = 3, kick angle α_k fixed, PSIO optimized in START
ITERP	1069	Size of Newton-Raphson iteration minus two
ITERPD	1060	ITERPD = 0, kick angle optimized; ITERPD = 1, kick angle fixed
JCOAST(6)	1130	Array for determining proper equations for terminating phases in COAST
JCOST	1129	Control parameter for COAST
JDATA	925	JDATA = 1, Newton-Raphson unconverged; JDATA = 2, Newton-Raphson converged; JDATA = 3, Newton-Raphson converged, but a propellant limitation has been exceeded
JFINAL(6)	1136	Array for determining proper equations for optimizing phases in FINAL

Variable	Common location	Definition
JKICK	1063	Index expediting selection of kick angles for interpolation
K	1182	Used in STEP to indicate converged and nonconverger output
LAST	711	Number of phases being integrated at a given time; may be equal to 1, LAST1, or LAST2
LAST1	753	Number of booster segments
LAST2	890	Number of upper phases plus one
MASH	1064	MASH = 0, problem proceeding normally; MASH = 1, if set to one in MAIN, then spacing too large; if set to one elsewhere, either the perturbation or change in initial conditions is so large as to prevent internal optimization
MODE(3)	1160	See section UPPER-PHASE DATA
MODEB(2)	1172	See section BOOSTER DATA
MODEC	1180	See section BOOSTER DATA
MODOUT	714	See section OUTPUT CONTROL
MODS	712	See section OUTPUT CONTROL
NCUTE	893	NCUTE = 0, last phase not terminated on energy; NCUTE = 1, last phase terminated on energy
NDAMP	926	Number of damping operations allowed in iteration (see section PROGRAM CONTROL VARIABLES)
NDUMP	1057	See section PROGRAM CONTROL VARIABLES
NEQ	709	Maximum index in integration loop
NFINAL	879	See section FINAL CONDITIONS
NOPT(6)	819	Indicates fixed or variable phase duration (see section UPPER-PHASE DATA)
NOPTA	1070	Number of first optimized phase
NOUT(6, 3)	1183	See section OUTPUT CONTROL
NPRINT	1181	NPRINT = 0, perturbation omitted; NPRINT = 1, perturbation printed
NSAVE(20)	1241	See subroutine COEFNT

Variable	Common location	Definition
NSETS(20)	1221	See section DRAG MODEL
NSHOT	929	NSHOT $\neq$ 0, a single upper-stage trajectory is integrated
NST	708	Minimum index in integration loop
NSTAGE	710	Phase counter during integration
NTB(3)	1116	Phase numbers of phases to be optimized in Newton-Raphson iteration
NVAR	1073	Number of parameters stored in each table entry
OBLAT(3)	780	Acceleration due to oblateness in X-, Y-, Z-directions
OBLATD	816	Fourth spherical harmonic coefficient in oblate earth model
OBLATH	817	Third spherical harmonic coefficient in oblate earth model
OBLATJ	818	Second spherical harmonic coefficient in oblate earth model
OBLATN	815	OBLATN = 0, spherical earth model; OBLATN = 1, oblate model utilized
OMEGA	405 605	Angular velocity
P	906	Semilatus rectum
PA	806	Pressure
PAR(6, 2)	1142	Partials of booster burnout conditions as function of α_k and booster burning time
PERB(2)	1055	See section BOOSTER DATA
PERIOD	910	Period of orbit
PHI	403 603	Polar angle traversed during upper phases
PHII	911	Inertial travel angle during booster
PHIR	912	Relative travel angle during booster
PROP(6)	849	Propellant sensitive structure factors
PSI	885	Thrust angle ψ

Variable	Common location	Definition
PSIDO	882	Initial $\dot{\psi}$ at beginning of variational phase
PSIO	881	Initial ψ at beginning of variational phase
PUSH	751	Thrust during booster
Q	807	Dynamic pressure
QVAL	808	QVAL = Q * AREA/W * G
R	402 602	Radius
RANGE	914	Surface distance from launch site to projection of position on planet
RB(5)	754	RB(1) X-component of radius vector; RB(2) Y-component of radius vector; RB(3) Z-component of radius vector; RB(4) = RB(1)**2 + RB(2)**2 + RB(3)**2; RB(5) = $\sqrt{\text{RB(4)}}$ = radius
RDOT	1155	Vertical velocity stored in table
RE	924	Radius of oblate earth
RERUN	927	See section BOOSTER DATA
RESERV	862	See section FINAL CONDITION
REVOLV	799	Angular rotation rate of planet
RMASS	401	Instantaneous weight during upper phases
RO	904	Reference radius for spherical earth, used for altitude calculations
ROA	1062	Reference radius used to improve interpolation accuracy for radius
S(6, 2)	1074	Parameters used in optimum staging equations
SINA(4)	791	SINA(J) = SIN(ANGLES(J)), J = 1, 4
SPSI	887	Sin ψ
STEP(3)	1166	See section UPPER-PHASE DATA
STEPB(2)	1176	See section BOOSTER DATA
STEPMX	705	Maximum number of integration steps allowed
STEPS	704	See section OUTPUT CONTROL

Variable	Common location	Definition
TB(6)	825	Phase duration list (see section UPPER-PHASE DATA)
TBOOST(6)	745	Phase durations for multiphase booster
TBURN	1065	Booster burning time
THETA	909	True anomaly
THRUST(6)	831	Thrusts (see section UPPER-PHASE DATA)
TIME	409 609	Time from launch
TKICK	1059	α_k, kick angle
TKTIME	804	Vertical rise time
TMIN	704	See section OUTPUT CONTROL
TMINB(2)	1178	See section BOOSTER DATA
TMIN1(3)	1169	See section UPPER-PHASE DATA
TOL(5, 2)	1044	Tolerances for iteration convergence and spacing (see section PROGRAM CONTROL DATA
TP	907	Time of perigee
TPD	908	Time of perigee departure
TRAVLI	913	Inertial distance traveled over planet surface during booster
TS(6)	932	Phasing times
TSTART	796	Time at beginning of booster integration
U	404 604	$\dot{R}$
UT(3)	800	Unit vector in direction of launch radius vector
V	889	Velocity in upper phases
V(6)	1154	Array containing altitude, vertical velocity, and horizontal velocity (a) from interpolation, (b) information stored in table
V(100, 2)	1301	See section DRAG MODEL; V(1, 1) = VARIND; V(1, 2) = VARDEP

Variable	Common location	Definition
VATM(5)	764	Relative velocity vector VATM(1) = V_{R_x}; VATM(2) = V_{R_y} VATM(3) = V_{R_2}; VATM(4) = VATM(1)**2 + VATM(2)**2 + VATM(3)**2; VATM(5) = $\sqrt{\mathrm{VATM(4)}}$
VELEX	870	Jet velocity
VELEXP(6)	864	Jet velocities for list of phases
VELSD	810	Velocity of sound
VMACH	811	Mach number
VSTART	797	Initial velocity at lift-off
VX(5)	759	VX(1) = V_x; VX(2) = V_y; V(3) = V_z; VX(4) = VX(1)**2 + VX(2)**2 + VX(3)**2; VX(5) = $\sqrt{\mathrm{VX(4)}}$
WEIGHT	402	Instantaneous weight during booster phase
WPMAX(6)	855	Maximum propellants allowed in six phases
WTFLOW(6)	733	Flows during booster phase (see section BOOSTER DATA)
WTO	720	Initial gross weight
X(100)	401	Working set of integration variables
XDOT(100)	501	Array of integration derivatives
XINPT(100)	601	Initial values of variables being integrated in variational phases
XO(6, 5)	1001	Array containing initial conditions for reference and perturbed trajectories in Newton-Raphson iteration
XPRIM(100, 2)	001	Two 100-variable double-precision arrays; second is integrated and first contains values of integration variables for last good step (see table I)
ZINCLI	919	Inertial inclination of trajectory
ZINCLR	920	Relative inclination of trajectory
ZLAM0	883	λ_0, scale factor, (see ref. 1)
ZLAM1	406 606	λ_1, Lagrange multiplier (see ref. 1)

Variable	Common location	Definition
ZLAM2	407 607	λ_2, Lagrange multiplier (see ref. 1)
ZLAM3	408 608	λ_3, Lagrange multiplier (see ref. 1)
ZLAM4	884	λ_4, Lagrange multiplier
ZLAT	916	Latitude
ZLONG	915	Longitude
ZNODEI	923	Longitude of ascending node (inertial)

TABLE I. - ELEMENTS OF INTEGRATION VARIABLE ARRAY XPRIM

[XPRIM 10 to 100 are left for expansion.]

Integration variables	XPRIM								
	1	2	3	4	5	6	7	8	9
Rectangular variables	Heat	Weight	$\dot{X}$ V_x	$\dot{Y}$ V_y	$\dot{Z}$ V_z	X R_x	Y R_y	Z R_z	Time
Polar variables	RMASS (weight)	R (radius)	PHI (polar angle)	U $\dot{R}$	OMEGA (angular velocity)	λ_1 (Lagrange multiplier)	λ_2 (Lagrange multiplier)	λ_3 (Lagrange multiplier	Time

APPENDIX E

EQUIVALENCE TABLE

THERE ARE FIFTY-TWO FORTRAN IV ROUTINES IN THIS CODE. THESE ARE LISTED IN ALPHABETIC ORDER. FOLLOWING THIS LIST IS A COMPILATION OF THE EQUIVALENCED VARIABLES IN THE CODE BY LOCATION, SEPARATED INTO THE THREE MAJOR COMMON BLOCKS.

LIST OF ROUTINES

1. AERO
2. ATMOS
3. ARCTAN
4. BOOST
5. CHECK
6. COAST
7. COEFNT
8. CONEVL
9. CONVT
10. DAMODE
11. DATAB
12. DATATM
13. DETERM
14. DOT
15. EQUAT1
16. EQUAT2
17. ERRORZ
18. FINAL
19. GUESS
20. INVERT
21. ITERAT
22. LOAD
23. MAIN
24. MAINA
25. MAINB
26. OBLATE
27. ODDMOD
28. ORBEL
29. ORBEL2
30. OUTPT1
31. OUTPT2
32. PERDAT
33. PERTB
34. QUAD
35. RENDER
36. RUNGEK
37. SADDA
38. SADDB
39. SCOMP
40. SETUP
41. SIMPRO
42. SORTXY
43. SSTAGE
44. STAGE
45. START
46. STDATA
47. STEP
48. STGSS
49. THRUST
50. TUDES
51. XOLOAD
52. ZMIN

COMMON RUNG

VARIABLE	SUB BLOCK	LOCATION	DIMENSION	SUBROUTINE	NO
RELERR	RUN	001	100	ERRORZ	17
XINC	RUN	001	100	ERRORZ	17
XINC	RUN	001	100	RUNGEK	36
XINC	RUN	001		STEP	47
A1	RUN	101		ERRORZ	17
A1	RUN	101		RUNGEK	36
A1	RUN	101		STEP	47
A2	RUN	102		ERRORZ	17
A2	RUN	102		RUNGEK	36
A2	RUN	102		STEP	47
DEL1	RUN	103		RUNGEK	36
DEL1	RUN	103		STEP	47
DELSTO	RUN	104		RUNGEK	36

DELSTO	RUN	104	STEP	47
E2	RUN	105	ERRORZ	17
E2	RUN	105	RUNGEK	36
E2	RUN	105	STEP	47
H2	RUN	106	COAST	6
H2	RUN	106	RUNGEK	36
H2	RUN	106	STEP	47
I	RUN	107	COAST	6
I	RUN	107	RUNGEK	36
I	RUN	107	STEP	47
NSTEP1	RUN	108	RUNGEK	36
NSTEP1	RUN	108	STEP	47
NSTEP2	RUN	109	RUNGEK	36
NSTEP2	RUN	109	STEP	47
NSTEP3	RUN	110	RUNGEK	36
NSTEP3	RUN	110	STEP	47
RATIO	RUN	111	RUNGEK	36
RATIO	RUN	111	STEP	47
STEPGO	RUN	112	ERRORZ	17
STEPNO	RUN	112	OUTPT1	30
STEPGO	RUN	112	OUTPT2	31
STEPGO	RUN	112	RUNGEK	36
STEPGO	RUN	112	STEP	47
STEPNO	RUN	113	OUTPT2	31
STEPNO	RUN	113	RUNGEK	36
STEPNO	RUN	113	STEP	47
SCRIBE	RUN	114	RUNGEK	36
SCRIBE	RUN	114	STEP	47
NSTAG1	RUN	115	OUTPT1	30
NSTAG1	RUN	115	RUNGEK	36
NSTAG1	RUN	115	STAGE	44
NSTAG1	RUN	115	STEP	47

COMMON ATABLE

VARIABLE	SUB BLOCK	LOCATION	DIMENSION	SUBROUTINE	NO
VEL	CME	0001		MAINB	25
FIRST	CME	0001	28	RENDER	35
VEL	CME	0001		TUDES	50
OBLATN	CME	0002		ATMOS	2
OBLATN	CME	0002		EQUAT2	16
OBLATN	CME	0002		MAINB	25
APR	CME	0004		ATMOS	2
A	CME	0004		MAINB	25
A	CME	0004		TUDES	50
ALT	CME	0005		MAINB	25
ALT	CME	0005		TUDES	50
XSTORO	CME	0006	4	MAINB	25
XSTORO	CME	0006	4	ORBEL2	29
TSPM	CME	0010		MAINA	24
TSPM	CME	0010		MAINB	25
TSPM	CME	0010		START	45
WSTORE	CME	0011		MAINB	25

HSTORE	CME	0012		MAINB	25
RADIUS	CME	0013		MAINB	25
RADIUS	CME	0013		TUDES	50
FUELT	CME	0014		MAINB	25
FUELT	CME	0014		START	45
HARDBT	CME	0015		MAINB	25
HARDBT	CME	0015		START	45
DELTB	CME	0016		BOOST	4
DELTB	CME	0016		MAINA	24
DELTB	CME	0016		MAINB	25
TBO	CME	0017		MAINB	25
TBLAST	CME	0018		MAINA	24
TBLAST	CME	0018		MAINB	25
TMINST	CME	0019		BOOST	4
TMINST	CME	0019		MAINA	24
TMINST	CME	0019		MAINB	25
NKICK	CME	0020		MAINB	25
NKICK	CME	0020		OUTPT2	31
NKICKS	CME	0021		MAINA	24
NKICKS	CME	0021		MAINB	25
UN	CME	0022	3	THRUST	49
UN	CME	0022	3	TUDES	50
ANGLEB	CME	0025	4	MAINB	25
ANGLEB	CME	0025	4	TUDES	50
DELTK	CME	0199		BOOST	4
DELTK	CME	0199		MAIN	23
DELTK	CME	0199		MAINA	24
SECOND	CME	0199	2	RENDER	35
IKICK	CME	0200		DATAB	11
IKICK	CME	0200		LOAD	22
IKICK	CME	0200		MAINA	24
IKICK	CME	0200		MAINB	25
IKICK	CME	0200		RENDER	35
IKICK	CME	0200		START	45
APT	CME	0201	3,100	MAINA	24
KPT	CME	0201	3,100	MAINA	24
THIRD	CME	0201	300	RENDER	35
VAR	CME	0501		MAINA	24
VAR	CME	0501		OUTPT2	31
FOURTH	CME	0501	7500	RENDER	35

COMMON CSTAR

VARIABLE	SUB BLOCK	LOCATION	DIMENSION	SUBROUTINE	NO
XPRIM	CMA	0001	100,2	MAIN	23
XPRIM	CMA	0001	100,2	MAINB	25
XPRIM	CMA	0001	100,2	RUNGEK	36
XPRIM	CMA	0001	100,2	STAGE	44
XPRIM	CMA	0001	100,2	STEP	47
XPRIM	CMA	0001	100,2	TUDES	50
RMASS	CMA	0003		TUDES	50
TIME	CMA	0017		MAINB	25

TIME	CMA	0017		SETUP	40
RMASS	CMA	0201		STAGE	44
WEIGHT	CMA	0203		STAGE	44
RMASS	CMA	0401		COAST	6
RMASS	CMA	0401		EQUAT1	15
X	CMA	0401	100	EQUAT2	16
RMASS	CMA	0401		ERRORZ	17
RMASS	CMA	0401		FINAL	18
X	CMA	0401	100	MAINB	25
X	CMA	0401		ORBEL2	29
RMASS	CMA	0401		OUTPT1	30
HEAT	CMA	0401		OUTPT2	31
X	CMA	0401	100	RUNGEK	36
X	CMA	0401	100	SCOMP	39
X	CMA	0401	100	STAGE	44
X	CMA	0401		STEP	47
WEIGHT	CMA	0402		AERO	1
R	CMA	0402		COAST	6
R	CMA	0402		CONEVL	8
R	CMA	0402		EQUAT1	15
R	CMA	0402		ERRORZ	17
WEIGHT	CMA	0402		ERRORZ	17
R	CMA	0402		FINAL	18
R	CMA	0402		MAIN	23
R	CMA	0402		ORBEL	28
R	CMA	0402		OUTPT1	30
WEIGHT	CMA	0402		OUTPT2	31
R	CMA	0402		PERTB	33
R	CMA	0402		SCOMP	39
R	CMA	0402		SSTAGE	43
R	CMA	0402		STAGE	44
WEIGHT	CMA	0402		THRUST	49
PHI	CMA	0403		EQUAT1	15
PHI	CMA	0403		ORBEL	28
PHI	CMA	0403		OUTPT1	30
U	CMA	0404		COAST	6
U	CMA	0404		CONEVL	8
U	CMA	0404		EQUAT1	15
U	CMA	0404		FINAL	18
U	CMA	0404		ORBEL	28
U	CMA	0404		OUTPT1	30
U	CMA	0404		SCOMP	39
U	CMA	0404		SSTAGE	43
U	CMA	0404		STAGE	44
OMEGA	CMA	0405		COAST	6
OMEGA	CMA	0405		CONEVL	8
OMEGA	CMA	0405		EQUAT1	15
OMEGA	CMA	0405		FINAL	18
OMEGA	CMA	0405		ORBEL	28
OMEGA	CMA	0405		OUTPT1	30
OMEGA	CMA	0405		SCOMP	39
OMEGA	CMA	0405		SSTAGE	43
OMEGA	CMA	0405		STAGE	44
ZLAM1	CMA	0406		COAST	6
ZLAM1	CMA	0406		CONEVL	8
ZLAM1	CMA	0406		EQUAT1	15
ZLAM1	CMA	0406		ERRORZ	17
ZLAM1	CMA	0406		FINAL	18
ZLAM1	CMA	0406		OUTPT1	30
ZLAM1	CMA	0406		SCOMP	39

ZLAM1	CMA	0406		SSTAGE	43
ZLAM2	CMA	0407		COAST	6
ZLAM2	CMA	0407		CONEVL	8
ZLAM2	CMA	0407		EQUAT1	15
ZLAM2	CMA	0407		ERRORZ	17
ZLAM2	CMA	0407		FINAL	18
ZLAM2	CMA	0407		OUTPT1	30
ZLAM2	CMA	0407		SCOMP	39
ZLAM2	CMA	0407		SSTAGE	43
ZLAM3	CMA	0408		COAST	6
ZLAM3	CMA	0408		CONEVL	8
ZLAM3	CMA	0408		EQUAT1	15
ZLAM3	CMA	0408		ERRORZ	17
ZLAM3	CMA	0408		FINAL	18
ZLAM3	CMA	0408		SCOMP	39
ZLAM3	CMA	0408		SSTAGE	43
TIME	CMA	0409		COAST	6
TIME	CMA	0409		EQUAT1	15
TIME	CMA	0409		ORBEL	28
TIME	CMA	0409		ORBEL2	29
TIME	CMA	0409		OUTPT1	30
TIME	CMA	0409		OUTPT2	31
TIME	CMA	0409		STEP	47
DRMASS	CMA	0501		EQUAT1	15
XDOT	CMA	0501	100	EQUAT2	16
XDOT	CMA	0501	100	RUNGEK	36
XDOT	CMA	0501	100	STEP	47
DR	CMA	0502		EQUAT1	15
DPHI	CMA	0503		EQUAT1	15
DU	CMA	0504		EQUAT1	15
DOMEGA	CMA	0505		EQUAT1	15
DZLAM1	CMA	0506		EQUAT1	15
DZLAM1	CMA	0506		OUTPT1	30
DZLAM2	CMA	0507		EQUAT1	15
DZLAM2	CMA	0507		OUTPT1	30
DZLAM3	CMA	0508		EQUAT1	15
DTIME	CMA	0509		DATAB	11
XINPT	CMA	0601	100	MAIN	23
XINPT	CMA	0601	100	PERDAT	32
XINPT	CMA	0601	100	PERTB	33
XINPT	CMA	0601	100	START	45
R	CMA	0602		START	45
U	CMA	0604		START	45
OMEGA	CMA	0605		START	45
ZLAM1	CMA	0606		START	45
ZLAM2	CMA	0607		START	45
ZLAM3	CMA	0608		START	45
TIME	CMA	0609		DAMODE	10
DELT	CMA	0701		COAST	6
DELT	CMA	0701		DAMODE	10
DELT	CMA	0701		ERRORZ	17
DELT	CMA	0701		MAINB	25
DELT	CMA	0701		RUNGEK	36
DELT	CMA	0701		SETUP	40
DELT	CMA	0701		STEP	47
DELMAX	CMA	0702		DAMODE	10
DELMAX	CMA	0702		MAINA	24
DELMAX	CMA	0702		RUNGEK	36
DELMAX	CMA	0702		SETUP	40
DELMAX	CMA	0702		STEP	47

TMIN	CMA	0703	DAMODE	10
TMIN	CMA	0703	MAINA	24
TMIN	CMA	0703	RUNGEK	36
TMIN	CMA	0703	SETUP	40
TMIN	CMA	0703	STEP	47
STEPS	CMA	0704	DAMODE	10
STEPS	CMA	0704	MAINA	24
STEPS	CMA	0704	RUNGEK	36
STEPS	CMA	0704	SETUP	40
STEPS	CMA	0704	STEP	47
STEPMX	CMA	0705	DATAB	11
STEPMX	CMA	0705	RUNGEK	36
STEPMX	CMA	0705	STEP	47
ERLIMT	CMA	0706	DATAB	11
ERLIMT	CMA	0706	RUNGEK	36
ERLIMT	CMA	0706	STEP	47
ERLOG	CMA	0707	MAIN	23
ERLOG	CMA	0707	RUNGEK	36
ERLOG	CMA	0707	STEP	47
NST	CMA	0708	DATAB	11
NST	CMA	0708	MAIN	23
NST	CMA	0708	RUNGEK	36
NST	CMA	0708	STEP	47
NEQ	CMA	0709	DATAB	11
NEQ	CMA	0709	ERRORZ	17
NEQ	CMA	0709	MAIN	23
NEQ	CMA	0709	RUNGEK	36
NEQ	CMA	0709	STAGE	44
NEQ	CMA	0709	START	45
NEQ	CMA	0709	STEP	47
NSTAGE	CMA	0710	COAST	6
NSTAGE	CMA	0710	DAMODE	10
NSTAGE	CMA	0710	EQUAT1	15
NSTAGE	CMA	0710	MAIN	23
NSTAGE	CMA	0710	MAINB	25
NSTAGE	CMA	0710	OUTPT1	30
NSTAGE	CMA	0710	RUNGEK	36
NSTAGE	CMA	0710	SETUP	40
NSTAGE	CMA	0710	SIMPRO	41
NSTAGE	CMA	0710	STAGE	44
NSTAGE	CMA	0710	START	45
NSTAGE	CMA	0710	STEP	47
LAST	CMA	0711	DATAB	11
LAST	CMA	0711	MAINA	24
LAST	CMA	0711	MAINB	25
LAST	CMA	0711	RUNGEK	36
LAST	CMA	0711	START	45
LAST	CMA	0711	STEP	47
MODS	CMA	0712	DAMODE	10
MODS	CMA	0712	RUNGEK	36
MODS	CMA	0712	SETUP	40
MODS	CMA	0712	STEP	47
EREF	CMA	0713	DATAB	11
EREF	CMA	0713	MAIN	23
MODOUT	CMA	0714	COAST	6
MODOUT	CMA	0714	DAMODE	10
MODOUT	CMA	0714	MAINA	24
MODOUT	CMA	0714	RUNGEK	36
MODOUT	CMA	0714	SETUP	40
MODOUT	CMA	0714	STEP	47

FM	CMA	0715		COAST	6
FM	CMA	0715		CONEVL	8
FM	CMA	0715		DATAB	11
FM	CMA	0715		EQUAT1	15
GM	CMA	0715		EQUAT2	16
FM	CMA	0715		FINAL	18
GM	CMA	0715		OBLATE	26
FM	CMA	0715		ORBEL	28
FM	CMA	0715		ORBEL2	29
FM	CMA	0715		OUTPT2	31
FM	CMA	0715		SCOMP	39
FM	CMA	0715		SSTAGE	43
FM	CMA	0715		STAGE	44
FM	CMA	0715		START	45
GM	CMA	0715		TUDES	50
FM	CMA	0715		XOLOAD	51
G	CMA	0716		AERO	1
G	CMA	0716		COAST	6
G	CMA	0716		CONEVL	8
G	CMA	0716		DATAB	11
G	CMA	0716		EQUAT1	15
G	CMA	0716		MAIN	23
G	CMA	0716		OUTPT1	30
G	CMA	0716		OUTPT2	31
G	CMA	0716		SSTAGE	43
G	CMA	0716		STAGE	44
G	CMA	0716		START	45
G	CMA	0716		THRUST	49
CONM	CMA	0717		ATMOS	2
CONM	CMA	0717		DATAB	11
CONM	CMA	0717		XOLOAD	51
CONN	CMA	0718		DATAB	11
CONN	CMA	0718		XOLOAD	51
CPA	CMA	0719		ATMOS	2
CPA	CMA	0719		XOLOAD	51
WTO	CMA	0720		MAINB	25
WTO	CMA	0720		START	45
HARDB	CMA	0721	6	MAINB	25
HARDB	CMA	0721	6	PERDAT	32
HARDB	CMA	0721	6	STAGE	44
EXITS	CMA	0727	6	PERDAT	32
EXITS	CMA	0727	6	SIMPRO	41
WTFLOW	CMA	0733	6	MAIN	23
WTFLOW	CMA	0733	6	MAINB	25
WTFLOW	CMA	0733	6	PERDAT	32
WTFLOW	CMA	0733	6	SIMPRO	41
FORCES	CMA	0739	6	MAIN	23
FORCES	CMA	0739	6	PERDAT	32
FORCES	CMA	0739	6	SIMPRO	41
TBOOST	CMA	0745	6	MAINA	24
TBOOST	CMA	0745	6	MAINB	25
TBOOST	CMA	0745	6	PERDAT	32
TBOOST	CMA	0745	6	START	45
THRUST	CMA	0751		OUTPT2	31
PUSH	CMA	0751		SIMPRO	41
PUSH	CMA	0751		THRUST	49
FLOW	CMA	0752		EQUAT2	16
FLOW	CMA	0752		OUTPT2	31
FLOW	CMA	0752		SIMPRO	41
LAST1	CMA	0753		DATAB	11

LAST1	CMA	0753		MAIN	23
LAST1	CMA	0753		MAINA	24
LAST1	CMA	0753		MAINB	25
LAST1	CMA	0753		START	45
RB	CMA	0754	5	AERO	1
RB	CMA	0754	5	ATMOS	2
RB	CMA	0754	5	EQUAT2	16
RB	CMA	0754	5	MAINB	25
RB	CMA	0754	5	OBLATE	26
RB	CMA	0754	5	ORBEL2	29
RB	CMA	0754	5	OUTPT2	31
RB	CMA	0754	5	THRUST	49
RBSQ	CMA	0757		ATMOS	2
R	CMA	0758		ATMOS	2
RB5	CMA	0758		ERRORZ	17
RADIUS	CMA	0758		OUTPT2	31
VX	CMA	0759	5	AERO	1
VX	CMA	0759	5	EQUAT2	16
VX	CMA	0759	5	ORBEL2	29
VX	CMA	0759	5	OUTPT2	31
VX5	CMA	0763		ERRORZ	17
VATM	CMA	0764	5	AERO	1
VATM	CMA	0764	5	EQUAT2	16
VATM	CMA	0764	5	ORBEL2	29
VATM	CMA	0764	5	OUTPT2	31
VATM	CMA	0764	5	THRUST	49
VAT5	CMA	0768		MAINB	25
H	CMA	0769	5	ORBEL2	29
H	CMA	0769	5	OUTPT2	31
FORCE	CMA	0774	3	PERDAT	32
FORCE	CMA	0774	3	STAGE	44
FORCE	CMA	0774	3	THRUST	49
DRAG	CMA	0777	3	AERO	1
DRAG	CMA	0777	3	PERDAT	32
DRAG	CMA	0777	3	STAGE	44
DRAG	CMA	0777	3	THRUST	49
OBLAT	CMA	0780	3	OBLATE	26
OBLAT	CMA	0780	3	PERDAT	32
OBLAT	CMA	0780	3	STAGE	44
OBLAT	CMA	0780	3	THRUST	49
COMPA	CMA	0783	3	EQUAT2	16
COMPA	CMA	0783	3	PERDAT	32
COMPA	CMA	0783	3	STAGE	44
COMPA	CMA	0783	3	THRUST	49
ANGLES	CMA	0786	4	DATAB	11
ANGLES	CMA	0786	4	MAINB	25
ANGLES	CMA	0786	4	TUDES	50
ELEV	CMA	0790		MAINA	24
ELEV	CMA	0790		MAINB	25
ELEV	CMA	0790		START	45
ELEV	CMA	0790		TUDES	50
SINA	CMA	0791	4	MAINB	25
SINA	CMA	0791	4	TUDES	50
ESTART	CMA	0795		DATAB	11
ESTART	CMA	0795		MAINB	25
TSTART	CMA	0796		DATAB	11
TSTART	CMA	0796		MAINB	25
VSTART	CMA	0797		DATAB	11
VSTART	CMA	0797		MAINB	25
ASTART	CMA	0798		DATAB	11

ASTART	CMA	0798		MAINB	25
REVOLV	CMA	0799		AERO	1
REVOLV	CMA	0799		DATAB	11
REVOLV	CMA	0799		MAINB	25
REVOLV	CMA	0799		ORBEL2	29
REVOLV	CMA	0799		TUDES	50
UT	CMA	0800	3	THRUST	49
UT	CMA	0800	3	TUDES	50
DELTBT	CMA	0803		DATAB	11
DELTBT	CMA	0803		MAINB	25
DELTBT	CMA	0803		SETUP	40
TKTIME	CMA	0804		DATAB	11
TKTIME	CMA	0804		MAINB	25
TKTIME	CMA	0804		TUDES	50
ALT	CMA	0805		ATMOS	2
ALTE	CMA	0805		MAINB	25
ALTE	CMA	0805		ORBEL2	29
ALTE	CMA	0805		OUTPT2	31
ALT	CMA	0805		STEP	47
PA	CMA	0806		AERO	1
PA	CMA	0806		ATMOS	2
PA	CMA	0806		OUTPT2	31
PA	CMA	0806		PERDAT	32
PA	CMA	0806		SIMPRO	41
PA	CMA	0806		STAGE	44
Q	CMA	0807		AERO	1
Q	CMA	0807		EQUAT2	16
Q	CMA	0807		OUTPT2	31
QVAL	CMA	0808		AERO	1
QVAL	CMA	0808		OUTPT2	31
QVAL	CMA	0808		PERDAT	32
QVAL	CMA	0808		STAGE	44
AREA	CMA	0809		AERO	1
VELSD	CMA	0810		AERO	1
VELSD	CMA	0810		ATMOS	2
VMACH	CMA	0811		AERO	1
VMACH	CMA	0811		OUTPT2	31
VMACH	CMA	0811		PERDAT	32
VMACH	CMA	0811		STAGE	44
CD	CMA	0812		AERO	1
CD	CMA	0812		OUTPT2	31
CD	CMA	0812		PERDAT	32
CD	CMA	0812		STAGE	44
A	CMA	0813		DATAB	11
STOA	CMA	0813		MAINB	25
A	CMA	0813		OBLATE	26
B	CMA	0814		ATMOS	2
B	CMA	0814		DATAB	11
B	CMA	0814		MAINB	25
B	CMA	0814		TUDES	50
OBLATN	CMA	0815		DATAB	11
OBLATS	CMA	0815		MAINB	25
OBLATN	CMA	0815		TUDES	50
OBLATD	CMA	0816		DATAB	11
OBLATD	CMA	0816		OBLATE	26
OBLATH	CMA	0817		DATAB	11
OBLATH	CMA	0817		OBLATE	26
OBLATJ	CMA	0818		DATAB	11
OBLATJ	CMA	0818		OBLATE	26
OBLATJ	CMA	0818		TUDES	50

NOPT	CMA	0819	6	CHECK	5
NOPT	CMA	0819	6	COAST	6
NOPT	CMA	0819	6	MAIN	23
NOPT	CMA	0819	6	PERDAT	32
NOPT	CMA	0819	6	START	45
NOPT	CMA	0819	6	XOLOAD	51
TB	CMA	0825	6	CHECK	5
TB	CMA	0825	6	COAST	6
TB	CMA	0825	6	MAINA	24
TB	CMA	0825	6	OUTPT1	30
TB	CMA	0825	6	PERDAT	32
TB	CMA	0825	6	START	45
TB	CMA	0825	6	XOLOAD	51
THRUST	CMA	0831	6	COAST	6
THRUST	CMA	0831	6	MAIN	23
THRUST	CMA	0831	6	OUTPT1	30
THRUST	CMA	0831	6	PERDAT	32
THRUST	CMA	0831	6	STAGE	44
THRUST	CMA	0831	6	START	45
THRUST	CMA	0831	6	XOLOAD	51
FLOMX	CMA	0837	6	CHECK	5
FLOMX	CMA	0837	6	COAST	6
FLOMX	CMA	0837	6	FINAL	18
FLOMX	CMA	0837	6	MAIN	23
FLOMX	CMA	0837	6	OUTPT1	30
FLOMX	CMA	0837	6	PERDAT	32
FLOMX	CMA	0837	6	SCOMP	39
FLOMX	CMA	0837	6	STAGE	44
FLOMX	CMA	0837	6	START	45
HARD	CMA	0843	6	COAST	6
HARD	CMA	0843	6	OUTPT1	30
HARD	CMA	0843	6	PERDAT	32
HARD	CMA	0843	6	STAGE	44
HARD	CMA	0843	6	START	45
PROP	CMA	0849	6	COAST	6
PROP	CMA	0849	6	FINAL	18
PROP	CMA	0849	6	OUTPT1	30
PROP	CMA	0849	6	PERDAT	32
PROP	CMA	0849	6	STAGE	44
PROP	CMA	0849	6	START	45
PROP	CMA	0849	6	XOLOAD	51
WPMAX	CMA	0855	6	CHECK	5
WPMAX	CMA	0855	6	PERDAT	32
DELTAV	CMA	0861		COAST	6
DELTAV	CMA	0861		FINAL	18
DELTAV	CMA	0861		OUTPT1	30
DELTAV	CMA	0861		PERDAT	32
DELTAV	CMA	0861		STAGE	44
RESERV	CMA	0862		OUTPT1	30
RESERV	CMA	0862		PERDAT	32
RESERV	CMA	0862		STAGE	44
DROP	CMA	0863		OUTPT1	30
DROP	CMA	0863		PERDAT	32
DROP	CMA	0863		STAGE	44
VELEXP	CMA	0864	6	MAIN	23
VELEXP	CMA	0864	6	OUTPT1	30
VELEXP	CMA	0864	6	STAGE	44
VELEXP	CMA	0864	6	START	45
VELEX	CMA	0870		COAST	6
VELEX	CMA	0870		EQUAT1	15

VELEX	CMA	0870		FINAL	18
VELEX	CMA	0870		MAIN	23
VELEX	CMA	0870		OUTPT1	30
VELEX	CMA	0870		STAGE	44
VELEX	CMA	0870		START	45
FUEL	CMA	0871	6	CHECK	5
FUEL	CMA	0871	6	COAST	6
FUEL	CMA	0871	6	OUTPT1	30
FUEL	CMA	0871	6	STAGE	44
FUEL	CMA	0871	6	START	45
FUEL	CMA	0871	6	XOLOAD	51
FLOW	CMA	0877		COAST	6
FLOW	CMA	0877		EQUAT1	15
FLOW	CMA	0877		OUTPT1	30
FLOW	CMA	0877		STAGE	44
FLOW	CMA	0877		START	45
FUELDV	CMA	0878		CHECK	5
FUELDV	CMA	0878		OUTPT1	30
FUELDV	CMA	0878		STAGE	44
NFINAL	CMA	0879		DATAB	11
NFINAL	CMA	0879		FINAL	18
NFINAL	CMA	0879		SCOMP	39
NFINAL	CMA	0879		XOLOAD	51
BETA	CMA	0880		FINAL	18
BETA	CMA	0880		PERDAT	32
BETA	CMA	0880		SCOMP	39
PSIO	CMA	0881		START	45
PSIO	CMA	0881		XOLOAD	51
PSIDO	CMA	0882		START	45
PSIDO	CMA	0882		XOLOAD	51
ZLAMO	CMA	0883		DATAB	11
ZLAMO	CMA	0883		START	45
ZLAM4	CMA	0884		COAST	6
ZLAM4	CMA	0884		CONEVL	8
ZLAM4	CMA	0884		DATAB	11
ZLAM4	CMA	0884		SSTAGE	43
ZLAM4	CMA	0884		START	45
PSI	CMA	0885		EQUAT1	15
PSI	CMA	0885		OUTPT1	30
FDM	CMA	0886		EQUAT1	15
FDM	CMA	0886		OUTPT1	30
SPSI	CMA	0887		EQUAT1	15
SPSI	CMA	0887		STAGE	44
CPSI	CMA	0888		EQUAT1	15
CPSI	CMA	0888		STAGE	44
V	CMA	0889		COAST	6
V	CMA	0889		EQUAT1	15
V	CMA	0889		ERRORZ	17
V	CMA	0889		FINAL	18
V	CMA	0889		ORBEL	28
V	CMA	0889		OUTPT1	30
V	CMA	0889		SCOMP	39
LAST	CMA	0890		CHECK	5
LAST	CMA	0890		COAST	6
LAST	CMA	0890		FINAL	18
LAST	CMA	0890		MAIN	23
LAST	CMA	0890		OUTPT1	30
LAST	CMA	0890		SCOMP	39
LAST	CMA	0890		STAGE	44
LAST2	CMA	0890		START	45

LAST	CMA	0890		XOLOAD	51
CAPPA	CMA	0891		COAST	6
CAPPA	CMA	0891		OUTPT1	30
CAPPA	CMA	0891		PERDAT	32
ENERGY	CMA	0892		COAST	6
ENERGY	CMA	0892		FINAL	18
ENERGY	CMA	0892		PERDAT	32
ENERGY	CMA	0892		SCOMP	39
ENERGY	CMA	0892		XOLOAD	51
NCUTE	CMA	0893		COAST	6
NCUTE	CMA	0893		FINAL	18
NCUTE	CMA	0893		SCOMP	39
NCUTE	CMA	0893		XOLOAD	51
CONST	CMA	0894	5,2	CONEVL	8
CONST	CMA	0894	5,2	FINAL	18
CONST	CMA	0894	5,2	MAIN	23
CONST	CMA	0894	5,2	SCOMP	39
RO	CMA	0904		DATAB	11
RO	CMA	0904		OUTPT1	30
E	CMA	0905		ORBEL	28
E	CMA	0905		ORBEL2	29
E	CMA	0905		OUTPT1	30
E	CMA	0905		OUTPT2	31
P	CMA	0906		ORBEL	28
P	CMA	0906		ORBEL2	29
P	CMA	0906		OUTPT1	30
P	CMA	0906		OUTPT2	31
TP	CMA	0907		ORBEL	28
TP	CMA	0907		ORBEL2	29
TP	CMA	0907		OUTPT1	30
TP	CMA	0907		OUTPT2	31
TPD	CMA	0908		ORBEL	28
TPD	CMA	0908		ORBEL2	29
TPD	CMA	0908		OUTPT1	30
TPD	CMA	0908		OUTPT2	31
THETA	CMA	0909		ORBEL	28
THETA	CMA	0909		ORBEL2	29
THETA	CMA	0909		OUTPT1	30
THETA	CMA	0909		OUTPT2	31
PERIOD	CMA	0910		DATAB	11
PERIOD	CMA	0910		ORBEL	28
PERIOD	CMA	0910		ORBEL2	29
PERIOD	CMA	0910		OUTPT1	30
PERIOD	CMA	0910		OUTPT2	31
PHII	CMA	0911		ORBEL2	29
PHII	CMA	0911		OUTPT2	31
PHIR	CMA	0912		ORBEL2	29
PHIR	CMA	0912		OUTPT2	31
TRAVLI	CMA	0913		ORBEL2	29
TRAVLI	CMA	0913		OUTPT2	31
RANGE	CMA	0914		ORBEL2	29
RANGE	CMA	0914		OUTPT2	31
ZLONG	CMA	0915		ORBEL2	29
ZLONG	CMA	0915		OUTPT2	31
ZLAT	CMA	0916		ORBEL2	29
ZLAT	CMA	0916		OUTPT2	31
BETAI	CMA	0917		ORBEL2	29
BETAI	CMA	0917		OUTPT2	31
BETAR	CMA	0918		ORBEL2	29
BETAR	CMA	0918		OUTPT2	31

ZINCLI	CMA	0919		ORBEL2	29
ZINCLI	CMA	0919		OUTPT2	31
ZINCLR	CMA	0920		ORBEL2	29
ZINCLR	CMA	0920		OUTPT2	31
AZIMI	CMA	0921		ORBEL2	29
AZIMI	CMA	0921		OUTPT2	31
AZIMR	CMA	0922		ORBEL2	29
AZIMR	CMA	0922		OUTPT2	31
ZNODEI	CMA	0923		ORBEL2	29
ZNODEI	CMA	0923		OUTPT2	31
RE	CMA	0924		ORBEL2	29
RE	CMA	0924		OUTPT2	31
JDATA	CMA	0925		CHECK	5
JDATA	CMA	0925		MAIN	23
JDATA	CMA	0925		STAGE	44
NDAMP	CMA	0926		DATAB	11
NDAMP	CMA	0926		MAIN	23
RERUN	CMA	0927		DATAB	11
RERUN	CMA	0927		MAIN	23
CMAX	CMA	0928		DATAB	11
CMAX	CMA	0928		MAIN	23
NSHOT	CMA	0929		DATAB	11
NSHOT	CMA	0929		MAIN	23
CLEAR	CMA	0930		DATAB	11
CLEAR	CMA	0930		MAIN	23
DELTST	CMA	0931		DAMODE	10
DELTST	CMA	0931		DATAB	11
DELTST	CMA	0931		MAIN	23
TS	CMA	0932	6	COAST	6
TS	CMA	0932	6	DAMODE	10
TS	CMA	0932	6	EQUAT1	15
TS	CMA	0932	6	MAINB	25
TS	CMA	0932	6	OUTPT1	30
TS	CMA	0932	6	SETUP	40
TS	CMA	0932	6	START	45
TS	CMA	0932	6	STEP	47
DX	CMA	0938	5	MAIN	23
DX	CMA	0938	5	STDATA	46
FY	CMA	0943	6,6	FINAL	18
FY	CMA	0943	6,6	MAIN	23
FY	CMA	0943	6,6	PERTB	33
XO	CMB	1001	6,5	MAIN	23
XO	CMB	1001	6,5	START	45
XO	CMB	1001	6,5	XOLOAD	51
COMP	CMB	1031	5	MAIN	23
COMP	CMB	1031	5	SCOMP	39
FYD	CMB	1036	5	FINAL	18
FYD	CMB	1036	5	PERDAT	32
FYD	CMB	1036	5	SCOMP	39
FYD	CMB	1036	5	XOLOAD	51
ERR	CMB	1041		DATAB	11
ERR	CMB	1041		MAIN	23
ERROR	CMB	1042	2	DATAB	11
ERROR	CMB	1042	2	MAIN	23
TOL	CMB	1044	5,2	DATAB	11
TOL	CMB	1044	5,2	MAIN	23
ERRMXK	CMB	1054		COAST	6
ERRMXK	CMB	1054		DATAB	11
ERRMXK	CMB	1054		MAIN	23
PERB	CMB	1055	2	DATAB	11

PERB	CMB	1055	2	MAINB	25
NDUMP	CMB	1057		DATAB	11
NDUMP	CMB	1057		MAIN	23
DELTKT	CMB	1058		DATAB	11
DELTKT	CMB	1058		MAIN	23
DELTKT	CMB	1058		MAINA	24
TKICK	CMB	1059		MAINA	24
TKICK	CMB	1059		OUTPT1	30
TKICK	CMB	1059		START	45
TKICK	CMB	1059		XOLOAD	51
ITERPD	CMB	1060		PERDAT	32
ITERPD	CMB	1060		START	45
ITERPD	CMB	1060		XOLOAD	51
IMODE	CMB	1061		DATAB	11
IMODE	CMB	1061		EQUAT1	15
IMODE	CMB	1061		ERRORZ	17
IMODE	CMB	1061		MAIN	23
IMODE	CMB	1061		MAINA	24
IMODE	CMB	1061		MAINB	25
IMODE	CMB	1061		OUTPT1	30
IMODE	CMB	1061		OUTPT2	31
IMODE	CMB	1061		SETUP	40
IMODE	CMB	1061		STAGE	44
IMODE	CMB	1061		START	45
IMODE	CMB	1061		STEP	47
IMODE	CMB	1061		THRUST	49
IMODE	CMB	1061		TUDES	50
ROA	CMB	1062		BOOST	4
ROA	CMB	1062		DATAB	11
ROA	CMB	1062		OUTPT2	31
ROA	CMB	1062		START	45
JKICK	CMB	1063		DATAB	11
JKICK	CMB	1063		MAIN	23
JKICK	CMB	1063		MAINA	24
JKICK	CMB	1063		MAINB	25
MASH	CMB	1064		COAST	6
MASH	CMB	1064		MAIN	23
MASH	CMB	1064		MAINA	24
MASH	CMB	1064		PERDAT	32
MASH	CMB	1064		START	45
MASH	CMB	1064		STEP	47
TBURN	CMB	1065		BOOST	4
TBURN	CMB	1065		MAINA	24
TBURN	CMB	1065		START	45
IKCKST	CMB	1066		MAINB	25
IKCKST	CMB	1066		RENDER	35
ITERAD	CMB	1067		START	45
ITERAD	CMB	1067		XOLOAD	51
ITER	CMB	1068		START	45
ITER	CMB	1068		XOLOAD	51
ITERP	CMB	1069		MAIN	23
ITERP	CMB	1069		XOLOAD	51
NOPTA	CMB	1070		COAST	6
NOPTA	CMB	1070		FINAL	18
NOPTA	CMB	1070		SCOMP	39
NOPTA	CMB	1070		XOLOAD	51
FIXDTK	CMB	1071		MAIN	23
FIXDTK	CMB	1071		OUTPT2	31
FIXDTK	CMB	1071		START	45
FIXDTK	CMB	1071		XOLOAD	51

IBURN	CMB	1072		LOAD	22
IBURN	CMB	1072		MAINA	24
IBURN	CMB	1072		STEP	47
NVAR	CMB	1073		DATAB	11
NVAR	CMB	1073		MAINA	24
NVAR	CMB	1073		MAINB	25
NVAR	CMB	1073		OUTPT2	31
NVAR	CMB	1073		RENDER	35
S	CMB	1074	6,2	COAST	6
S	CMB	1074	6,2	FINAL	18
S	CMB	1074	6,2	SADDA	37
S	CMB	1074	6,2	SADDB	38
S	CMB	1074	6,2	SCOMP	39
S	CMB	1074	6,2	STAGE	44
S	CMB	1074	6,2	START	45
IDATA	CMB	1086	6,5	COAST	6
IDATA	CMB	1086	6,5	FINAL	18
IDATA	CMB	1086	6,5	SCOMP	39
IDATA	CMB	1086	6,5	START	45
IDATA	CMB	1086	6,5	XOLOAD	51
NTB	CMB	1116	4	START	45
NTB	CMB	1116	4	XOLOAD	51
ITK	CMB	1120	3	BOOST	4
ITK	CMB	1120	3	MAINA	24
ITB	CMB	1123	3	BOOST	4
ITB	CMB	1123	3	MAINA	24
APTMAX	CMB	1126	3	BOOST	4
APTMAX	CMB	1126	3	MAINA	24
JCOST	CMB	1129		COAST	6
JCOST	CMB	1129		MAINB	25
JCOST	CMB	1129		STAGE	44
JCOST	CMB	1129		START	45
JCOAST	CMB	1130	6	COAST	6
JCOAST	CMB	1130	6	START	45
JCOAST	CMB	1130	6	XOLOAD	51
JFINAL	CMB	1136	6	FINAL	18
JFINAL	CMB	1136	6	SCOMP	39
JFINAL	CMB	1136	6	XOLOAD	51
PAR	CMB	1142	6,2	BOOST	4
PAR	CMB	1142	6,2	START	45
V	CMB	1154	6	BOOST	4
V	CMB	1154	6	LOAD	22
ALTB	CMB	1154		OUTPT2	31
V	CMB	1154	6	START	45
RDOT	CMB	1155		ORBEL2	29
RDOT	CMB	1155		OUTPT2	31
HORV	CMB	1156		OUTPT2	31
MODE	CMB	1160	3	DAMODE	10
MODE	CMB	1160	3	DATAB	11
DELMX	CMB	1163	3	DAMODE	10
DELMX	CMB	1163	3	DATAB	11
STEP	CMB	1166	3	DAMODE	10
STEP	CMB	1166	3	DATAB	11
TMIN1	CMB	1169	3	DAMODE	10
TMIN1	CMB	1169	3	DATAB	11
MODEB	CMB	1172	2	DATAB	11
MODEB	CMB	1172	2	SETUP	40
DELMXB	CMB	1174	2	DATAB	11
DELMXB	CMB	1174	2	SETUP	40
STEPB	CMB	1176	2	DATAB	11

STEPB	CMB	1176	2	SETUP	40
TMINB	CMB	1178	2	DATAB	11
TMINB	CMB	1178	2	SETUP	40
MODEC	CMB	1180		DATAB	11
MODEC	CMB	1180		MAINA	24
MODEC	CMB	1180		OUTPT2	31
NPRINT	CMB	1181		DAMODE	10
NPRINT	CMB	1181		DATAB	11
K	CMB	1182		DAMODE	10
K	CMB	1182		SETUP	40
K	CMB	1182		STEP	47
NOUT	CMB	1183	6,3	PERDAT	32
NOUT	CMB	1183	6,3	STEP	47
ICC	CMB	1201	20	DATAB	11
ICC	CMB	1201	20	QUAD	34
NSETS	CMB	1221	20	COEFNT	7
NSETS	CMB	1221	20	PERDAT	32
NSAVE	CMB	1241	20	COEFNT	7
NSAVE	CMB	1241	20	DATAB	11
V	CMB	1301	100,2	COEFNT	7
COEFN	CMB	1501	500	COEFNT	7
COEFN	CMB	1501	500	QUAD	34

APPENDIX F

TABLE FOR INPUT ROUTINE

```
$TABLE,
 813 =A      , 880 =ANG    , 809 =AREA   , 798 =ASTART, 788 =AZMUTH,
 814 =B      , 930.=CLEAR  , 928.=CMAX   ,1501 =COEFN , 717 =CONM  ,
 718 =CONN   , 861 =DELTAV, 803 =DELTB  ,1058 =DELTK  ,1163 =DELMX ,
1174 =DELMXB, 863 =DROP   , 931 =DELTST, 938 =DX     , 892 =ENERGY,
 713 =EREF   , 706 =ERLIMT,1041 =ERR    ,1054 =ERRMXK,1042 =ERROR ,
 727 =EXITB  , 727 =EXITS  , 837 =FLOW   , 733 =FLOWB  , 715 =FM    ,
 739 =FORCES,1036 =FYD    , 716 =G      , 843 =HARD   , 721 =HARDB ,
1201.=ICC    ,1061.=IMODE  ,1060.=ITERPD, 753.=LAST   , 786 =LAT   ,
 787 =LONG   ,1160.=MODE   ,1172.=MODEB  ,1180.=MODEC  , 926.=NDAMP ,
1057.=NDUMP  , 879.=NFINAL, 819.=NOPT   ,1181.=NPRINT,1195.=NPRNTB,
1189.=NPRNTC,1183.=NPRNTI,1241.=NSAVE  ,1221.=NSETS  , 815 =OBLATN,
1055 =PERB   , 849 =PROP   , 881 =PSI    , 882 =PSID   , 927.=RERUN ,
 862 =RESERV, 799 =REVOLV, 904 =RO     ,1062 =ROA    ,1166.=STEP  ,
1176.=STEPB  , 705.=STEPMX, 825 =TB     , 745 =TBOOST, 831 =THRUST,
1059 =TKICK  , 804 =TKTIME,1169 =TMIN   ,1178 =TMINB  ,1044 =TOL   ,
 796 =TSTART,1401 =VARDEP,1301 =VARIND, 797 =VSTART, 855 =WPMAX ,
 720 =WTO    , 884 =ZLAM4 /
```

APPENDIX G

SAMPLE PROBLEM

```
      AREA=855                                               ,$$ REFERENCE AREA FOR DRAG
      ICC=1                                                  ,$$ DRAG COEFFICIENT LOCATION
      NSAVE=1                                                ,$$ COEFFICIENT LOCATUN
      NSFTS=10                                               ,$$
      VARIND=0,0.07,0.15,0.30,0.50,0.57,0.65,0.88,1.12,1.17,1.25,1.50,2.00,2.50,$$
      3.00,3.50,4.00,5.00,6.00,7.00,8.00,$$
      VARDEP=1.15,1.15,1.15,1.15,0.80,0.45,0.44,0.46,0.70,0.90,0.92,0.90,0.76,0.56,0.45,$$
      0.40,0.38,0.36,0.34,0.32,0.31,0.30,$$
      LAT=28.3                                               ,$$ LATITUDE
      LONG=279.5                                             ,$$ LONGITUDE
      AZMUTH=90                                              ,$$ AZIMUTH HEADING
      WTO=6.00E+06                                           ,$$ INITIAL GROSS WEIGHT
      EXITS=7.8964431E+04/144                                ,$$ ENGINE EXIT AREA
      RERUN=1                                                ,$$ FLAG FOR CHECKING SPACING
      MODEB=51,51                                            ,$$ BOOSTER OUTPUT INSTRJCTION
      MODEC=2                                                ,$$ BOOSTER OUTPUT TYPE
      PERB=0.6,0.8                                           ,$$ TABLE DELT SPACING
      TKTIME=15                                              ,$$ VERTICAL RISE TIME
      TB=142,250,500                                         ,$$ GUESS AT PHASE DURATIONS
      THRUST=8.6647727E+06,1.5E+06,2.5E+05                   ,$$ THRUSTS FOR THE PHASES
      FLOW=THRUST/305,THRUST(2)/428,THRUST(3)/850            ,$$ FLOW FOR THE PHASES
      PROP=0.03,0.033,0.12                                   ,$$ STRUCTURE FACTORS
      HARD=245000,70000,35000                                ,$$ FIXED HARDWARE WEIGHTS
      NOPT=1,1,1                                             ,$$ PHASE OPTIMIZATION FLAGS
      PSI=40/57.3                                            ,$$ GUESS AT PSIO
      PSID=-0.05/57.3                                        ,$$ GUESS AT PSIDO
      TKICK=89.4                                             ,$$ GUESS AT KICK ANGLE
      ITERPD=0                                               ,$$ KICK ANGLE IS OPTIMIZED
      MODE=61,52                                             ,$$ UPPER PHASE OUTPUT FLAGS
      NFINAL=1                                               ,$$ FINAL CONDITON TYPE FLAG
      FYD=121*6076.1155+A                                    ,$$ FINAL RADIUS
      ENERGY=-FM/FYD/2                                       ,$$ FINAL ENERGY
      ANG=0                                                  ,$$ FINAL FLIGHT PATH ANGLE
      DELTAV=0.414*25492.311                                 ,$$ DELTAV TO ESCAPE

COEFFICIENTS   1, 41
      0               1.14999998          0.34647418E-06     -0                0.15000000          1.57499994
     -3.08333337      1.66666684          0.50000000          1.26785628      -2.94523606          2.61904451
      0.65000000     -0.47401163          1.72756687         -0.44711671       1.11999999         -6.09991992
     11.8498634      -4.99994445          1.25000000          2.00000006      -1.14666656          0.21333338
      2.00000000      1.59999987         -0.75999999          0.11999999       3.00000000          0.52000010
     -0.40000021E-01 -0                   4.00000000          0.43999997      -0.19999996E-01     -0.18626451E-08
      6.00000000      0.37999982         -0.99999458E-02     -0.18626451E-08   80.0000000

 0           , 1.1500000    , 0.3464742E-06,-0             ,$$

 0.1500000   , 1.5749999    ,-3.0833334    , 1.6666668     ,$$

 0.5000000   , 1.2678563    ,-2.9452361    , 2.6190445     ,$$

 0.6500000   ,-0.4740116    , 1.7275669    ,-0.4471167     ,$$

 1.1200000   ,-6.0999199    , 11.849863    ,-4.9999444     ,$$

 1.2500000   , 2.0000001    ,-1.1466666    , 0.2133334     ,$$

 2.0000000   , 1.5999999    ,-0.7600000    , 0.1200000     ,$$

 3.0000000   , 0.5200001    ,-0.4000002E-01,-0             ,$$

 4.0000000   , 0.4400000    ,-0.2000000E-01,-0.1862645E-08,$$

 6.0000000   , 0.3799998    ,-0.9999946E-02,-0.1862645E-08,$$

 80.000000   ,
```

```
A1 THRUST ONE      THRUST FOUR      FLOW ONE      FLOW FOUR      HARD ONE          HARD FOUR      PROP ONE       PROP FOUR
A2 THRUST TWO      THRUST FIVE      FLOW TWO      FLOW FIVE      HARD TWO          HARD FIVE      PROP TWO       PROP FIVE
A3 THRUST THREE    THRUST SIX       FLOW THREE    FLOW SIX       HARD THREE        HARD SIX       PROP THREE     PROP SIX

U1 TIME            WEIGHT           PSI           ALTITUDE       ECCENTRICITY      INR. GAMMA     PERIOD         INR. VELOCITYY
U2 FLOW            ACCELLERATION    PSIO          PERIGEE ALT    TIME OF PERIGE    OMEGA          TRAVEL ANGLE   RAD. VELOCITY
U3 STEPNO          CAPPA            KICK ANGLE    RADIUS         TIME PER DEP      SEM LAT REC    TRUE ANOMALY   HOR. VELOCITY

U4 BURN TIME ONE   BURN TIME FOUR   WP ONE        WP FOUR        HARD ONE          HARD FOUR      M F ONE        M F FOUR
U5 BURN TIME TWO   BURN TIME FIVE   WP TWO        WP FIVE        HARD TWO          HARD FIVE      M F TWO        M F FIVE
U6 BURN TIME THRE  BURN TIME SIX    WP THREE      WP SIX         HARD THREE        HARD SIX       M F THREE      M F SIX

A1  8664772.6       0                28409.090     0               245000.00        0               0.3000000E-01  0
A2  1500000.0       0                3504.6729     0               70000.000        0               0.3300000E-01  0
A3  250000.00       0                294.11765     0               35000.000        0               0.1200000      0

B1 TIME            WEIGHT           RAD. VELOCITY  HOR. VELOCITY  ALTITUDE         INR. VELOCITY  INR. GAMMA     FLOW
B2 STEPGO+STEPNO   RADIUS           DRAG           PRESSURE       HEAT INTEGRAL    MACH NUMBER    Q              THRUST

B1  0                6000000.0       0              1342.5289     -0.25000000       1342.5289      0               28409.090
B2     0        0    0.2090990E 08   0              2124.1934      0                0              0               7499941.2

B1  15.000000        5573863.6       144.40608      1342.1591      1023.0000        1349.9053      6.1409652       28409.090
B2     4        0    0.2091092E 08   0              0              11523.263        0              23.286603       7540853.9
```

	B1 TIME B2 STEPGO+STEPNO		WEIGHT RADIUS	RAD. VELOCITY DRAG	HOR. VELOCITY PRESSURE	ALTITUDE HEAT INTEGRAL	INR. VELOCITY MACH NUMBER	INR. GAMMA Q	FLOW THRUST

KICK ANGLE = 89.299999

B1	15.000000		5573863.6	144.40224	1344.3592	1022.7500	1352.0924	6.1308300	28409.090
B2	0	0	0.2091092E 08	22896.708	2049.5845	11523.263	0.1274009	23.286762	7540854.0
B1	126.72000		2400000.1	1961.1498	6150.3975	120384.75	6455.5016	17.685689	28409.090
B2	37	9	0.2102999E 08	50416.089	9.9122334	0.9333217E 08	4.9994240	173.42412	8659337.0
B1	128.48000		2350000.1	1985.1508	6339.8041	123557.00	6643.3380	17.386640	28409.090
B2	38	9	0.2103346E 08	45830.513	8.5945016	0.9487445E 08	5.1403921	158.96856	8660059.5
B1	130.24000		2300000.1	2009.3107	6534.0762	127071.75	6836.0428	17.093322	28409.090
B2	39	9	0.2103698E 08	41648.050	7.4565729	0.9633811E 08	5.2833019	145.69618	8660683.6
B1	132.00000		2250000.1	2033.6892	6733.3775	130629.50	7033.7944	16.805947	28409.090
B2	40	9	0.2104053E 08	37841.519	6.4733821	0.9772675E 08	5.4286627	133.54111	8661222.7
B1	133.76000		2200000.1	2058.3493	6937.8853	134230.00	7236.7849	16.524702	28409.090
B2	41	9	0.2104414E 08	34382.914	5.6231947	0.9904409E 08	5.5770907	122.43246	8661689.0
B1	135.52000		2150000.1	2083.3582	7147.7925	137874.25	7445.2213	16.249753	28409.090
B2	42	9	0.2104778E 08	31241.105	4.8867957	0.1002939E 09	5.7292897	112.28555	8662092.9
B1	137.28000		2100000.1	2108.7867	7363.3087	141562.75	7659.3274	15.981246	28409.090
B2	43	9	0.2105147E 08	28385.530	4.2476648	0.1014798E 09	5.8860735	103.01480	8662443.2
B1	139.04000		2050000.1	2134.7105	7584.6611	145296.75	7879.3447	15.719310	28409.090
B2	44	9	0.2105520E 08	25824.189	3.6914306	0.1026051E 09	6.0483548	94.529479	8662748.4
B1	140.80000		2000000.1	2161.2089	7812.0947	149076.75	8105.5321	15.464057	28409.090
B2	45	9	0.2105898E 08	23571.500	3.2058692	0.1036729E 09	6.2171630	86.741782	8663014.5
B1	142.56000		1950000.1	2188.3676	8045.8785	152903.50	8338.1721	15.215588	28409.090
B2	46	9	0.2106281E 08	21500.807	2.7804829	0.1046858E 09	6.3936354	79.563523	8663247.9
B1	144.32000		1900000.1	2216.2779	8286.3051	156779.00	8577.5720	14.973992	28409.090
B2	47	9	0.2106669E 08	19533.849	2.4001935	0.1056453E 09	6.5784715	72.710023	8663456.4
B1	146.08000		1850000.1	2245.0371	8533.6914	160704.50	8824.0627	14.739348	28409.090
B2	48	9	0.2107061E 08	17973.468	2.0819939	0.1065572E 09	6.7992596	67.375303	8663630.9
B1	147.84000		1800000.1	2274.7486	8788.3810	164681.25	9078.0021	14.511724	28409.090
B2	49	9	0.2107459E 08	16500.708	1.8027265	0.1074296E 09	7.0268655	62.309057	8663784.0
B1	149.60000		1750000.1	2305.5251	9050.7531	168711.00	9339.7847	14.291184	28409.090
B2	50	9	0.2107862E 08	15114.208	1.5580119	0.1082626E 09	7.2616420	57.509351	8663918.2
B1	151.36000		1700000.1	2337.4876	9321.2218	172796.00	9609.8400	14.077786	28409.090
B2	51	9	0.2108271E 08	13812.046	1.3439028	0.1090563E 09	7.5039735	52.972257	8664035.6
B1	153.12000		1650000.1	2370.7667	9600.2397	176938.50	9888.6370	13.871583	28409.090
B2	52	9	0.2108685E 08	12659.563	1.1561005	0.1098118E 09	7.7810810	48.997455	8664138.6
B1	154.88000		1600000.1	2405.5023	9888.2991	181140.75	10176.684	13.672622	28409.090
B2	53	9	0.2109105E 08	11687.021	0.9905260	0.1105389E 09	8.1230402	45.751055	8664229.4
B1	156.64000		1550000.1	2441.8477	10185.945	185405.25	10474.545	13.480951	28409.090
B2	54	9	0.2109532E 08	10730.371	0.8439999	0.1112390E 09	8.4834207	42.518939	8664309.7
B1	158.40000		1500000.1	2479.9703	10493.780	189735.50	10782.841	13.296617	28409.090
B2	55	9	0.2109965E 08	9794.5067	0.7148964	0.1119096E 09	8.8638275	39.317305	8664380.5
B1	160.16000		1450000.1	2520.0529	10812.468	194134.50	11102.258	13.119667	28409.090
B2	56	9	0.2110405E 08	8884.3950	0.6016930	0.1125489E 09	9.2660542	36.163150	8664442.5
B1	161.92000		1400000.1	2562.2960	11142.745	198605.75	11433.554	12.950148	28409.090
B2	57	9	0.2110852E 08	8004.6801	0.5029601	0.1131549E 09	9.6921358	33.072766	8664496.7
B1	163.68000		1350000.1	2606.9198	11485.431	203153.50	11777.570	12.788111	28409.090
B2	58	9	0.2111307E 08	7159.5969	0.4173103	0.1137262E 09	10.144398	30.061431	8664543.7
B1	165.44000		1300000.1	2654.1683	11841.434	207781.75	12135.245	12.633610	28409.090
B2	59	9	0.2111770E 08	6353.2277	0.3434680	0.1142614E 09	10.625481	27.144481	8664584.2
B1	167.20000		1250000.1	2704.3119	12211.774	212495.75	12507.627	12.486703	28409.090
B2	60	9	0.2112241E 08	5589.0754	0.2802176	0.1147594E 09	11.138444	24.335619	8664618.9
B1	168.96000		1200000.0	2757.6522	12597.598	217301.00	12895.895	12.347454	28409.090
B2	61	9	0.2112722E 08	4870.3175	0.2264254	0.1152195E 09	11.686815	21.647982	8664648.4
B1	168.96000		1200000.0	2757.6522	12597.598	217301.00	12895.895	12.347454	28409.090
B2	61	9	0.2112722E 08	0	0	0.1152195E 09	0	21.647982	8664648.4

KICK ANGLE = 89.400000

B1	15.000000		5573863.6	144.40510	1344.1072	1022.7500	1351.8421	6.1320913	28409.090
B2	0	0	0.2091092E 08	22896.708	2049.5845	11523.263	0.1274009	23.286762	7540854.0
B1	126.72000		2400000.1	2455.5039	5742.0590	133466.00	6245.0573	23.153234	28409.090
B2	38	8	0.2104337E 08	26689.852	5.7926630	0.7379589E 08	4.7196608	90.322901	8661596.1
B1	128.48000		2350000.1	2502.0815	5921.0373	137828.50	6427.9931	22.907697	28409.090
B2	39	8	0.2104773E 08	23608.688	4.8953549	0.7456439E 08	4.8463988	80.486032	8662088.1
B1	130.24000		2300000.1	2549.4236	6104.7896	142273.50	6615.7401	22.665922	28409.090
B2	40	8	0.2105218E 08	20872.545	4.1352511	0.7527338E 08	4.9770542	71.704212	8662505.0
B1	132.00000		2250000.1	2597.6082	6293.4732	146802.25	6808.4780	22.428209	28409.090
B2	41	8	0.2105671E 08	18439.001	3.4894729	0.7592710E 08	5.1128572	63.853572	8662859.0

B1	133.76000		2200000.1	2646.7190	6487.2592	151416.75	7006.4009	22.194833	28409.090
B2	42	8	0.2106132E 08	16266.456	2.9385940	0.7552933E 08	5.2552059	56.808995	8663161.1
B1	135.52000		2150000.1	2696.8456	6686.3340	156119.00	7209.7183	21.966041	28409.090
B2	43	8	0.2106602E 08	14249.792	2.4582922	0.7708284E 08	5.4013765	50.204245	8663424.5
B1	137.28000		2100000.1	2748.0829	6890.8984	160939.75	7418.6549	21.742061	28409.090
B2	44	8	0.2107082E 08	12673.906	2.0665721	0.7759251E 08	5.5871594	45.157587	8663639.4
B1	139.04000		2050000.1	2800.5317	7101.1690	165792.25	7633.4514	21.523094	28409.090
B2	45	8	0.2107570E 08	11226.796	1.7316470	0.7806585E 08	5.7783476	40.472938	8663823.0
B1	140.80000		2000000.1	2854.3017	7317.3846	170768.00	7854.3719	21.309329	28409.090
B2	46	8	0.2108067E 08	9904.3752	1.4462331	0.7850378E 08	5.9751767	36.144135	8663979.5
B1	142.56000		1950000.1	2909.5099	7539.8038	175839.25	8081.7010	21.100938	28409.090
B2	47	8	0.2108575E 08	8758.9854	1.2031971	0.7890752E 08	6.1829611	32.197921	8664112.7
B1	144.32000		1900000.1	2966.2806	7768.7050	181009.50	8315.7440	20.898073	28409.090
B2	48	8	0.2109092E 08	7868.0894	0.9953663	0.7928285E 08	6.4724038	29.188528	8664226.7
B1	146.08000		1850000.1	3024.7484	8004.3931	186280.75	8556.8342	20.700879	28409.090
B2	49	8	0.2109619E 08	7009.5377	0.8163746	0.7963327E 08	6.7786284	26.258575	8664324.9
B1	147.84000		1800000.1	3085.0595	8247.2031	191656.50	8805.3365	20.509487	28409.090
B2	50	8	0.2110156E 08	6189.3086	0.6633697	0.7995783E 08	7.1032056	23.429470	8664408.7
B1	149.60000		1750000.1	3147.3711	8497.4993	197140.00	9061.6466	20.324021	28409.090
B2	51	8	0.2110705E 08	5412.8564	0.5336444	0.8025587E 08	7.4479151	20.721414	8664479.9
B1	151.36000		1700000.1	3211.8533	8755.6798	202735.50	9326.1960	20.144594	28409.090
B2	52	8	0.2111264E 08	4684.7665	0.4246129	0.8052706E 08	7.8148066	18.152135	8664539.7
B1	153.12000		1650000.1	3278.6899	9022.1794	208446.25	9599.4548	19.971312	28409.090
B2	53	8	0.2111836E 08	4009.0534	0.3338618	0.8077138E 08	8.2062078	15.738008	8664589.5
B1	154.88000		1600000.1	3348.0809	9297.4747	214277.00	9881.9373	19.804277	28409.090
B2	54	8	0.2112419E 08	3388.6408	0.2591069	0.8098912E 08	8.6248378	13.492082	8664630.5
B1	156.64000		1550000.1	3420.2439	9582.0894	220231.75	10174.208	19.643584	28409.090
B2	55	8	0.2113014E 08	2825.6840	0.1982380	0.8118093E 08	9.0738319	11.425267	8664663.9
B1	158.40000		1500000.1	3495.4170	9876.5994	226316.50	10476.887	19.489327	28409.090
B2	56	8	0.2113623E 08	2321.1644	0.1492897	0.8134775E 08	9.5569077	9.5447098	8664690.7
B1	160.16000		1450000.1	3573.8610	10181.642	232536.50	10790.658	19.341598	28409.090
B2	57	8	0.2114245E 08	1875.2396	0.1104759	0.8149078E 08	10.078420	7.8550763	8664712.0
B1	161.92000		1400000.1	3655.8628	10497.921	238897.00	11116.280	19.200486	28409.090
B2	58	8	0.2114881E 08	1487.0989	0.8017585E-01	0.8161152E 08	10.643536	6.3578971	8664728.5
B1	163.68000		1350000.1	3741.7397	10826.221	245405.50	11454.592	19.066083	28409.090
B2	59	8	0.2115532E 08	1154.8419	0.5692589E-01	0.8171169E 08	11.258523	5.0509226	8664741.4
B1	165.44000		1300000.1	3831.8430	11167.416	252069.25	11806.532	18.938483	28409.090
B2	60	8	0.2116198E 08	875.74802	0.3943109E-01	0.8179318E 08	11.930961	3.9290515	8664751.0
B1	167.20000		1250000.1	3926.5642	11522.483	258894.75	12173.147	18.817783	28409.090
B2	61	8	0.2116881E 08	646.32396	0.2655784E-01	0.8185803E 08	12.670091	2.9843530	8664758.0
B1	168.96000		1200000.0	4026.3414	11892.522	265891.50	12555.617	18.704083	28409.090
B2	62	8	0.2117581E 08	453.22121	0.1731022E-01	0.8190816E 08	13.311385	2.1470740	8664763.1
B1	168.96000		1200000.0	4026.3414	11892.522	265891.50	12555.617	18.704083	28409.090
B2	62	8	0.2117581E 08	0	0	0.8190816E 08	0	2.1470740	8664763.1

KICK ANGLE = 89.499999

B1	15.000000		5573863.6	144.40752	1343.8552	1022.7500	1351.5918	6.1333345	28409.090
B2	0	0	0.2091092E 08	22896.708	2049.5846	11523.253	0.1274009	73.286762	7540854.0
B1	126.72000		2400000.0	2950.5878	5227.4019	146336.25	6002.6411	29.442396	28409.090
B2	36	3	0.2105624E 08	14853.523	3.5507188	0.6083463E 08	4.4642369	49.534614	8662825.5
B1	128.48000		2350000.0	3021.2751	5390.6908	151591.00	6179.6158	29.268955	28409.090
B2	37	3	0.2106149E 08	12803.705	2.9196109	0.6123750E 08	4.5869120	42.999542	8663171.5
B1	130.24000		2300000.0	3093.4461	5558.5358	156971.50	6361.3465	29.096890	28409.090
B2	38	3	0.2106688E 08	10976.908	2.3835090	0.6159900E 08	4.7183731	37.144917	8663465.5
B1	132.00000		2250000.0	3167.1998	5731.0843	162480.50	6548.0137	28.926591	28409.090
B2	39	3	0.2107238E 08	9538.8228	1.9522803	0.6192452E 08	4.8833768	32.589723	8663702.0
B1	133.76000		2200000.1	3242.6420	5908.4962	168120.50	6739.8112	28.758422	28409.090
B2	40	3	0.2107802E 08	8244.3633	1.5916721	0.6221935E 08	5.0531233	28.449290	8663899.7
B1	135.52000		2150000.1	3319.8879	6090.9469	173395.00	6936.9510	28.592718	28409.090
B2	41	3	0.2108380E 08	7086.3380	1.2915150	0.6248498E 08	5.2278028	24.707903	8664054.4
B1	137.28000		2100000.1	3399.0601	6278.6254	179807.25	7139.6602	28.429790	28409.090
B2	42	3	0.2108971E 08	6149.8824	1.0406715	0.6272464E 08	5.4640543	21.749124	8664201.9
B1	139.04000		2050000.1	3480.2898	6471.7361	185860.50	7348.1824	28.269922	28409.090
B2	43	3	0.2109577E 08	5313.7916	0.8295351	0.6294325E 08	5.7362412	19.106784	8664317.6
B1	140.80000		2000000.1	3563.7194	6670.5027	192058.50	7562.7841	28.113384	28409.090
B2	44	3	0.2110196E 08	4537.9548	0.6530060	0.6314080E 08	6.0261805	16.599670	8664414.5
B1	142.56000		1950000.1	3649.5016	6875.1675	198405.25	7783.7518	27.960420	28409.090
B2	45	3	0.2110831E 08	3857.4215	0.5070645	0.6331723E 08	6.3357854	14.248271	8664494.5

B1	144.32000		1900000.1	3737.8004	7085.9935	204905.50	8011.3954	27.811260	28409.090
B2	46	3	0.2111481E 08	3233.3705	0.3878778	0.6347282E 08	6.6672870	12.069554	8664559.9
B1	146.08000		1850000.1	3828.7931	7303.2657	211563.25	8246.0513	27.666117	28409.090
B2	47	3	0.2112147E 08	2668.9806	0.2918545	0.6360814E 08	7.0232744	10.077288	8664612.5
B1	147.84000		1800000.1	3922.6705	7527.2981	218383.75	8488.0835	27.525190	28409.090
B2	48	3	0.2112829E 08	2166.0240	0.2156309	0.6372405E 08	7.4068170	8.2808009	8664654.4
B1	149.60000		1750000.1	4019.6394	7758.4256	225372.25	8737.8879	27.388665	28409.090
B2	49	3	0.2113528E 08	1724.9980	0.1561138	0.6382169E 08	7.8215533	6.6853700	8664687.0
B1	151.36000		1700000.1	4119.9237	7997.0225	232534.00	8995.8959	27.256718	28409.090
B2	50	3	0.2114244E 08	1345.1083	0.1104895	0.6390242E 08	8.2718366	5.2920369	8664712.0
B1	153.12000		1650000.1	4223.7669	8243.4906	239875.25	9262.5776	27.129513	28409.090
B2	51	3	0.2114978E 08	1024.2496	0.7622665E-01	0.6396780E 08	8.7629544	4.0973778	8664730.7
B1	154.88000		1600000.1	4331.4341	8498.2740	247402.75	9538.4478	27.007210	28409.090
B2	52	3	0.2115731E 08	759.15142	0.5108666E-01	0.6401951E 08	9.3013892	3.0938638	8664744.5
B1	156.64000		1550000.1	4443.2150	8761.8608	255123.50	9824.0708	26.889958	28409.090
B2	53	3	0.2116503E 08	545.52353	0.3312232E-01	0.6405934E 08	9.8951820	2.2702116	8664754.4
B1	158.40000		1500000.1	4559.4274	9034.7881	263044.25	10120.068	26.777904	28409.090
B2	54	3	0.2117295E 08	376.13078	0.2060534E-01	0.6408908E 08	10.538694	1.6019586	8664761.2
B1	160.16000		1450000.1	4680.4203	9317.6492	271174.25	10427.124	26.671189	28409.090
B2	55	3	0.2118109E 08	241.39282	0.1252986E-01	0.6410988E 08	10.898293	1.0417447	8664765.7
B1	161.92000		1400000.1	4806.5789	9611.1016	279521.50	10745.998	26.569954	28409.090
B2	56	3	0.2118943E 08	152.87716	0.7521651E-02	0.6412379E 08	11.271943	0.6689733	8664768.5
B1	163.68000		1350000.1	4938.3292	9915.8762	288095.25	11077.531	26.474336	28409.090
B2	57	3	0.2119801E 08	95.491111	0.4455010E-02	0.6413297E 08	11.660637	0.4240248	8664770.1
B1	165.44000		1300000.1	5076.1445	10232.788	296906.50	11422.662	26.384474	28409.090
B2	58	3	0.2120682E 08	58.790558	0.2601811E-02	0.6413895E 08	12.065480	0.2651325	8664771.1
B1	167.20000		1250000.1	5220.5534	10562.751	305965.25	11782.439	26.300509	28409.090
B2	59	3	0.2121588E 08	34.776480	0.1509280E-02	0.6414275E 08	12.220481	0.1577772	8664771.7
B1	168.96000		1200000.0	5372.1478	10906.791	315285.00	12158.045	26.222586	28409.090
B2	60	3	0.2122520E 08	20.646622	0.8908458E-03	0.6414508E 08	12.267275	0.9384190E-01	8664772.1
B1	168.96000		1200000.0	5372.1478	10906.791	315285.00	12158.045	26.222586	28409.090
B2	60	3	0.2122520E 08	0	0	0.6414508E 08	0	0.9384190E-01	8664772.1
B1	TIME		WEIGHT	RAD. VELOCITY	HOR. VELOCITY	ALTITUDE	INR. VELOCITY	INR. GAMMA	FLOW
B2	STEPGO+STEPNO		RADIUS	DRAG	PRESSURE	HEAT INTEGRAL	MACH NUMBER	Q	THRUST
B1	15.000000		5573863.6	144.40427	1344.1845	1022.7500	1351.9190	6.1317059	28409.090
B2	0	0	0.2091092E 08	22896.708	2049.5846	11523.263	0.1274009	23.286762	7540854.0
B1	141.86617		1969711.2	2666.5626	7620.9010	166942.00	8073.9514	19.285004	28409.090
B2	43	10	0.2107685E 08	0	0	0.8525801E 08	0	44.056735	8663861.7
U1	141.86617		1603802.7	34.722583	166958.75	0.9135463	19.285004	1952.9455	8073.9511
U2	3504.6729		0.9352771	-0.4094846E-01	-0.1995205E 08	-741.75146	0.2071683E-01	0	2666.5625
U3	0		0.9336385E-02	89.369293	0.2107685E 08	883.61763	1832855.0	178.09129	7620.9009
U1	395.58308		714607.88	27.269355	636969.25	0.5597554	6.8998464	2742.9387	17132.630
U2	3504.6729		2.0990532	-0.1498558E-01	-0.1479273E 08	-853.48028	0.4522785E-01	7.9106708	2058.2144
U3	19		-0	89.369293	0.2154686E 08	1249.0634	9541279.6	174.50667	17008.550
U1	395.58308		615264.45	27.269355	636969.25	0.5597554	6.8998464	2742.9387	17132.630
U2	294.11765		0.4063293	-0.1498558E-01	-0.1479273E 08	-853.48028	0.4522785E-01	7.9106708	2058.2144
U3	19		-0.1816063E 09	89.369293	0.2154686E 08	1249.0634	9541279.6	174.50667	17008.550
U1	999.28226		437705.86	19.518920	741988.00	0.8544922E-03	-0.4157721E-01	5338.8527	25502.984
U2	294.11765		0.5711598	-0.1591849E-01	732543.25	1879.1456	0.6748666E-01	41.171320	-18.505475
U3	34		4.0000000	89.369293	0.2165188E 08	-879.86337	0.2166093E 08	-60.072076	25502.978
U1	999.28226		224442.69	19.518920	741988.00	0.8544922E-03	-0.4157721E-01	5338.8527	25502.984
U2	294.11765		0.5711598	-0.1591849E-01	732543.25	1879.1456	0.6748666E-01	41.171320	-18.506475
U3	34		4.0000000	89.369293	0.2165188E 08	-879.86337	0.2166093E 08	-60.072076	25502.978
U4	141.86617		0	4030288.7	0	365908.66	0	0.9167670	0
U5	253.71691		0	889194.77	0	99343.427	0	0.8995047	0
U6	603.69919		0	317697.99	0	73123.759	0	0.8128974	0
B1	TIME		WEIGHT	RAD. VELOCITY	HOR. VELOCITY	ALTITUDE	INR. VELOCITY	INR. GAMMA	FLOW
B2	STEPGO+STEPNO		RADIUS	DRAG	PRESSURE	HEAT INTEGRAL	MACH NUMBER	Q	THRUST

KICK ANGLE = 89.349999

B1	15.000000		5573863.6	144.40372	1344.2332	1022.7500	1351.9673	6.1314628	28409.090
B2	0	0	0.2091092E 08	22896.708	2049.5845	11523.263	0.1274009	23.286762	7540854.0
B1	126.72000		2400000.0	2208.0924	5957.9770	126836.00	6353.9879	20.335265	28409.090
B2	40	10	0.2103674E 08	36450.029	7.5273920	0.8254204E 08	4.8578877	124.34763	8660644.7
B1	128.48000		2350000.0	2243.1953	6142.7394	130753.00	6539.5086	20.061098	28409.090
B2	41	10	0.2104066E 08	32666.899	6.4419537	0.8362368E 08	4.9906117	112.31123	8661240.0
B1	130.24000		2300000.0	2278.7472	6332.3339	134731.75	6729.8693	19.791723	28409.090
B2	42	10	0.2104464E 08	29268.523	5.5148907	0.8463519E 08	5.1259624	101.43452	8661748.4

B1	132.00000		2250000.0	2314.8167	6526.9209	138773.75	6925.2490	19.527396	28409.090
B2	43	10	0.2104868E 08	26217.521	4.7218099	0.8558089E 08	5.2647598	91.614383	8662183.2
B1	133.76000		2200000.0	2351.4770	6726.6745	142379.75	7125.8400	19.268347	28409.090
B2	44	10	0.2105278E 08	23477.232	4.0418899	0.8546487E 08	5.4079722	82.746840	8662556.1
B1	135.52000		2150000.1	2388.8057	6931.7850	147050.75	7331.8508	19.014783	28409.090
B2	45	10	0.2105696E 08	21011.155	3.4572631	0.8729093E 08	5.5567043	74.724849	8662876.7
B1	137.28000		2100000.1	2426.8858	7142.4579	151288.00	7543.5059	18.766888	28409.090
B2	46	10	0.2106119E 08	18783.756	2.9526993	0.8806243E 08	5.7121996	67.441000	8663153.4
B1	139.04000		2050000.1	2465.8062	7358.9184	155593.00	7761.0489	18.524829	28409.090
B2	47	10	0.2106550E 08	16658.750	2.5056029	0.8878142E 08	5.8677334	60.388054	8663398.6
B1	140.80000		2000000.1	2505.6609	7581.4085	159967.50	7984.7414	18.288757	28409.090
B2	48	10	0.2106987E 08	15044.213	2.1383274	0.8945199E 08	6.0673344	55.102093	8663600.0
B1	142.56000		1950000.1	2546.5485	7810.1877	164413.00	8214.8611	18.058802	28409.090
B2	49	10	0.2107432E 08	13601.066	1.8203228	0.9008376E 08	6.2728454	50.138992	8663774.4
B1	144.32000		1900000.1	2588.5772	8045.5434	168931.25	8451.7159	17.835086	28409.090
B2	50	10	0.2107884E 08	12258.996	1.5456405	0.9067723E 08	6.4845409	45.495143	8663925.0
B1	146.08000		1850000.1	2631.8634	8287.7896	173524.50	8695.6404	17.617722	28409.090
B2	51	10	0.2108343E 08	11015.236	1.3089433	0.9123310E 08	6.7027195	41.164321	8664054.7
B1	147.84000		1800000.1	2676.5313	8537.2670	178195.25	8946.9967	17.406808	28409.090
B2	52	10	0.2108810E 08	9969.4441	1.1042300	0.9175363E 08	6.9727106	37.580353	8664167.0
B1	149.60000		1750000.1	2722.7137	8794.3440	182945.75	9206.1749	17.202435	28409.090
B2	53	10	0.2109285E 08	9046.0739	0.9260077	0.9224748E 08	7.2904415	34.452465	8664264.7
B1	151.36000		1700000.1	2770.5557	9059.4281	187778.75	9473.6063	17.004688	28409.090
B2	54	10	0.2109769E 08	8150.4319	0.7709303	0.9271393E 08	7.6260566	31.384353	8664349.7
B1	153.12000		1650000.1	2820.2142	9332.9655	192697.75	9749.7616	16.813647	28409.090
B2	55	10	0.2110261E 08	7287.9609	0.6368161	0.9315160E 08	7.9811785	28.395277	8664423.4
B1	154.88000		1600000.1	2871.8592	9615.4443	197706.00	10035.156	16.629386	28409.090
B2	56	10	0.2110762E 08	6463.8030	0.5216078	0.9355939E 08	8.3576350	25.504038	8664486.5
B1	156.64000		1550000.1	2925.6759	9907.4022	202806.75	10330.353	16.451977	28409.090
B2	57	10	0.2111272E 08	5682.6287	0.4233603	0.9393648E 08	8.7574997	22.728378	8664540.4
B1	158.40000		1500000.1	2981.8657	10209.429	208004.50	10635.975	16.281489	28409.090
B2	58	10	0.2111792E 08	4948.3115	0.3402214	0.9428239E 08	9.1831641	20.083731	8664586.0
B1	160.16000		1450000.1	3040.6494	10522.177	213303.00	10952.705	16.117989	28409.090
B2	59	10	0.2112322E 08	4264.3092	0.2704719	0.9459693E 08	9.6373506	17.584723	8664624.2
B1	161.92000		1400000.1	3102.2696	10846.368	218707.75	11281.302	15.961547	28409.090
B2	60	10	0.2112862E 08	3633.1408	0.2124889	0.9488026E 08	10.123244	15.243111	8664656.0
B1	163.68000		1350000.1	3166.9936	11182.804	224223.25	11622.606	15.812233	28409.090
B2	61	10	0.2113414E 08	3056.8286	0.1647825	0.9513285E 08	10.644506	13.069535	8664682.2
B1	165.44000		1300000.1	3235.1175	11532.381	229855.25	11977.554	15.670119	28409.090
B2	62	10	0.2113977E 08	2536.4853	0.1259688	0.9535552E 08	11.205446	11.071830	8664703.5
B1	167.20000		1250000.1	3306.9704	11896.098	235611.00	12347.194	15.535282	28409.090
B2	63	10	0.2114553E 08	2072.3420	0.9477502E-01	0.9554939E 08	11.811190	9.2550601	8664720.6
B1	168.96000		1200000.0	3382.9203	12275.082	241496.50	12732.705	15.407806	28409.090
B2	64	10	0.2115141E 08	1664.0305	0.7005370E-01	0.9571588E 08	12.467759	7.6226484	8664734.1
B1	168.96000		1200000.0	3382.9203	12275.082	241496.50	12732.705	15.407806	28409.090
B2	64	10	0.2115141E 08	0	0	0.9571588E 08	0	7.6226484	8664734.1
B1	TIME		WEIGHT	RAD. VELOCITY	HOR. VELOCITY	ALTITUDE	INR. VELOCITY	INR. GAMMA	FLOW
B2	STEPGO+STEPNO		RADIUS	DRAG	PRESSURE	HEAT INTEGRAL	MACH NUMBER	Q	THRUST
B1	15.000000		5573863.6	144.40433	1344.1793	1022.7500	1351.9137	6.1317322	28409.090
B2	0	0	0.2091092E 08	22896.708	2049.5846	11523.263	0.1274009	23.286762	7540854.0
B1	141.86232		1969820.4	2681.4959	7609.2579	167403.00	8067.9135	19.412413	28409.090
B2	42	9	0.2107731E 08	0	0	0.8477190E 08	0	43.265638	8663876.7
U1	141.86232		1603915.1	34.643657	167419.75	0.9138139	19.412413	1952.7882	8067.9132
U2	3504.6729		0.9352116	-0.4095543E-01	-0.1995507E 08	-741.17240	0.2068473E-01	0	2681.4959
U3	0		0.9350344E-02	89.371392	0.2107731E 08	883.03473	1827338.7	178.08405	7609.2578
U1	395.70713		714272.10	27.202741	639858.75	0.5596458	6.9077446	2743.8466	17134.040
U2	3504.6729		2.1000400	-0.1484268E-01	-0.1478985E 08	-853.60007	0.4522475E-01	7.9094726	2060.7285
U3	19		-0.1164153E-09	89.371392	0.2154975E 08	1249.3072	9545090.6	174.49775	17009.665
U1	395.70713		614913.88	27.202741	639858.75	0.5596458	6.9077446	2743.8466	17134.040
U2	294.11765		0.4065610	-0.1484268E-01	-0.1478985E 08	-853.60007	0.4522475E-01	7.9094726	2060.7285
U3	19		-0.1814946E 09	89.371392	0.2154975E 08	1249.3072	9545090.6	174.49775	17009.665
U1	998.86699		437513.93	19.701302	748787.25	0.2441406E-03	-0.9717724E-02	5338.8527	25494.980
U2	294.11765		0.5714104	-0.1515999E-01	745763.75	1768.0430	0.6744432E-01	41.131941	-4.3241090
U3	34		8.0000000	89.371392	0.2165868E 08	-769.17600	0.2166094E 08	-58.359989	25494.980
U1	998.86699		224338.62	19.701302	748787.25	0.2441406E-03	-0.9717724E-02	5338.8527	25494.980
U2	294.11765		0.5714104	-0.1515999E-01	745763.75	1768.0430	0.6744432E-01	41.131941	-4.3241090
U3	34		8.0000000	89.371392	0.2165868E 08	-769.17600	0.2166094E 08	-58.359989	25494.980
U4	141.86232		0	4030179.6	0	365905.38	0	0.9167656	0
U5	253.84481		0	889643.02	0	99358.219	0	0.8995368	0
U6	603.15985		0	317477.91	0	73097.349	0	0.8128470	0

```
     TB=144                                          ,$$ FIRST STAGE DURATION
     $$    NOTE THAT THE FIRST STAGE IS FIXED IN DURATION
     NOPT=0,1,1                                      ,$$ PHASE OPTIMIZATION FLAGS
     MODEC=3                                         ,$$ BOOSTER OUTPUT TYPE

A1 THRUST ONE       THRUST FOUR      FLOW ONE         FLOW FOUR        HARD ONE         HARD FOUR        PROP ONE         PROP FOUR
A2 THRUST TWO       THRUST FIVE      FLOW TWO         FLOW FIVE        HARD TWO         HARD FIVE        PROP TWO         PROP FIVE
A3 THRUST THREE     THRUST SIX       FLOW THREE       FLOW SIX         HARD THREE       HARD SIX         PROP THREE       PROP SIX

U1 TIME             WEIGHT           PSI              ALTITUDE         ECCENTRICITY     INR. GAMMA       PERIOD           INR. VELOCITYY
U2 FLOW             ACCELLERATION    PSID             PERIGEE ALT      TIME OF PERIGE   OMEGA            TRAVEL ANGLE     RAD. VELOCITY
U3 STEPNO           CAPPA            KICK ANGLE       RADIUS           TIME PER DEP     SEM LAT REC      TRUE ANOMALY     HOR. VELOCITY

U4 BURN TIME ONE    BURN TIME FOUR   WP ONE           WP FOUR          HARD ONE         HARD FOUR        M F ONE          M F FOUR
U5 BURN TIME TWO    BURN TIME FIVE   WP TWO           WP FIVE          HARD TWO         HARD FIVE        M F TWO          M F FIVE
U6 BURN TIME THRE   BURN TIME SIX    WP THREE         WP SIX           HARD THREE       HARD SIX         M F THREE        M F SIX

A1  8664772.6        0                28409.090        0                245000.00        0                 0.3000000E-01   0
A2  1500000.0        0                3504.6729        0                70000.000        0                 0.3300000E-01   0
A3  250000.00        0                294.11765        0                35000.000        0                 0.1200000       0

B1 TIME             LATITUDE         RADIUS           INR. VELOCITY    PERIOD           SEM LAT REC      WEIGHT           ACCELLERATION
B2 STEPGO+STEPNO    LONGITUDE        ALTITUDE         INR. GAMMA       TRUE ANOMALY     ECCENTRICITY     FLOW             THRUST
B3 PSI              INR. AZIMUTH     RANGE            REL. VELOCITY    NODE             INR. INCLIN.     RAD. VELOCITY    INR TRVL ANGL
B4 PSID             REL. AZIMUTH     ALPHA            REL. GAMMA       TIME PER DEP     REL. INCLIN.     HOR VELOCITY     REL TRVL ANGL
B5 DRAG             PRESSURE         HEAT INTEGR.     MACH     NO      DRAG COEFF       DYNAM. PRES.     TIME OF PERIGE   INR.TRAVEL

B1   15.000000       28.299972        0.2091092E 08    1351.9360        1794.0381        55127.982        5573863.6        1.3487874
B2     0        0    279.49999        1022.7500        6.1316203        179.98343        0.9973159        28409.090        7540854.0
B3   89.362491       90.000000        0.8181382E-10    144.41302        189.56266        28.299972        144.40408        0.2638724E-03
B4 -1.3528310        89.999966        0                89.362491        891.56180        28.299971        1344.2018        0.1362145E-11
B5  22896.708        2049.5846        11523.263        0.1274009        1.1500000        23.286763       -876.56180        0.1584883E-05

B1  144.00000        28.292814        0.2108103E 08    8375.0155        19[illegible]5.0361      1988558.2        1909090.9        4.5382801
B2    42        8    280.24094        171119.50        18.624490        177.98967        0.9062285        28409.090        8663989.5
B3   22.112170       90.666348        0.1874056E-05    7105.5584        189.43693        28.300011        2674.6817        0.5553225E-07
B4 -0.1837770        90.803360        0                22.112170        888.57816        28.303274        7936.4326        0.3120175E-07
B5  0                0                0.8742585E 08    0                0                41.217055       -744.57816        0.3395469E-05

U1  144.00000        1541363.8        33.483757        171137.00        0.9062285        18.624490        1965.0360        8375.0154
U2  3504.6729        0.9731641       -0.3997838E-01   -0.1986670E 08   -744.57870        0.2157030E-01    0                2674.6817
U3    0             -4.0000000        89.362491        0.2108103E 08    888.57870        1988558.2        177.98967        7936.4325

U1  380.10155        713905.12        26.753526        631864.75        0.5639655        7.2217565        2732.7799        17068.319
U2  3504.6729        2.1011195       -0.1445424E-01   -0.1486619E 08   -859.60535        0.4503740E-01    7.4804338        2145.6577
U3    15            -4.0000000        89.362491        0.2154175E 08    1239.7069        9452133.0        174.34212        16932.917

U1  380.10155        616598.98        26.753526        631864.75        0.5639655        7.2217565        2732.7799        17068.319
U2  294.11765        0.4054499       -0.1445424E-01   -0.1486619E 08   -859.60535        0.4503740E-01    7.4804338        2145.6577
U3    15            -0.1828609E 09    89.362491        0.2154175E 08    1239.7069        9452133.0        174.34212        16932.917

U1  989.37902        437399.73        20.320702        751615.50        0.1726335E-03    0.1681029E-02    5338.8527        25491.652
U2  294.11765        0.5715596       -0.1124348E-01    747313.00        653.79829        0.6742671E-01    40.964771        0.7479122
U3    30            -0               89.362491        0.2166151E 08    335.58073        0.2166094E 08    131.58272        25491.652

U1  989.37902        224049.45        20.320702        751615.50        0.1726335E-03    0.1681029E-02    5338.8527        25491.652
U2  294.11765        0.5715596       -0.1124348E-01    747313.00        653.79829        0.6742671E-01    40.964771        0.7479122
U3    30            -0               89.362491        0.2166151E 08    335.58073        0.2166094E 08    131.58272        25491.652

U4  144.00000        0                4090909.0        0                367727.27        0                0.9175247        0
U5  236.10155        0                827458.69        0                97306.136        0                0.8947774        0
U6  609.27747        0                319240.65        0                73308.877        0                0.8132494        0

     TB=144,250                                      ,$$ FIRST TWO STAGE DURATIONS
     NOPT=0,0,1                                      ,$$ PHASE OPTIMIZATION FLAGS
     $$    NOTE THAT THE FIRST AND SECOND STAGES ARE FIXED IN DURATION

A1 THRUST ONE       THRUST FOUR      FLOW ONE         FLOW FOUR        HARD ONE         HARD FOUR        PROP ONE         PROP FOUR
A2 THRUST TWO       THRUST FIVE      FLOW TWO         FLOW FIVE        HARD TWO         HARD FIVE        PROP TWO         PROP FIVE
A3 THRUST THREE     THRUST SIX       FLOW THREE       FLOW SIX         HARD THREE       HARD SIX         PROP THREE       PROP SIX

U1 TIME             WEIGHT           PSI              ALTITUDE         ECCENTRICITY     INR. GAMMA       PERIOD           INR. VELOCITYY
U2 FLOW             ACCELLERATION    PSID             PERIGEE ALT      TIME OF PERIGE   OMEGA            TRAVEL ANGLE     RAD. VELOCITY
U3 STEPNO           CAPPA            KICK ANGLE       RADIUS           TIME PER DEP     SEM LAT REC      TRUE ANOMALY     HOR. VELOCITY

U4 BURN TIME ONE    BURN TIME FOUR   WP ONE           WP FOUR          HARD ONE         HARD FOUR        M F ONE          M F FOUR
U5 BURN TIME TWO    BURN TIME FIVE   WP TWO           WP FIVE          HARD TWO         HARD FIVE        M F TWO          M F FIVE
U6 BURN TIME THRE   BURN TIME SIX    WP THREE         WP SIX           HARD THREE       HARD SIX         M F THREE        M F SIX
```

A1	8664772.6		0	28409.090	0	245000.00	0	0.3000000E-01	0
A2	1500000.0		0	3504.6729	0	70000.000	0	0.3300000E-01	0
A3	250000.00		0	294.11765	0	35000.000	0	0.1200000	0
B1	TIME		LATITUDE	RADIUS	INR. VELOCITY	PERIOD	SEM LAT REC	WEIGHT	ACCELLERATION
B2	STEPGO+STEPNO		LONGITUDE	ALTITUDE	INR. GAMMA	TRUE ANOMALY	ECCENTRICITY	FLOW	THRUST
B3	PSI		INR. AZIMUTH	RANGE	REL. VELOCITY	NODE	INR. INCLIN.	RAD. VELOCITY	INR TRVL ANGL
B4	PSIO		REL. AZIMUTH	ALPHA	REL. GAMMA	TIME PER DEP	REL. INCLIN.	HOR VELOCITY	REL TRVL ANGL
B5	DRAG		PRESSURE	HEAT INTEGR.	MACH NO	DRAG COEFF	DYNAM. PRES.	TIME OF PERIGE	INR.TRAVEL
B1	15.000000		28.299972	0.2091092E 08	1351.9882	1794.0355	56132.374	5573863.6	1.3487874
B2	0	0	279.49999	1022.7500	6.1313572	179.98343	0.9973157	28409.090	7540854.0
B3	89.341623		90.000000	0.8181382E-10	144.41302	189.56266	28.299971	144.40348	0.2638724E-03
B4	-1.3527791		89.999977	0	89.341623	891.96212	28.299976	1344.2543	0.1362145E-11
B5	22896.708		2049.5846	11523.263	0.1274009	1.1500000	23.286763	-876.96212	0.1584883E-05
B1	144.00000		28.292661	0.2107600E 08	8429.8875	1966.4421	2042321.7	1909090.9	4.5381982
B2	44	11	280.25666	166087.75	17.382145	178.07620	0.9036066	28409.090	8663833.1
B3	20.623329		90.673302	0.1913838E-05	7149.9452	189.43799	28.300010	2518.3734	0.5719450E-07
B4	-0.1836811		90.809464	0	20.623329	894.63690	28.303282	8044.9237	0.3186410E-07
B5	0		0	0.9277741E 08	0	0	50.070842	-750.63690	0.3435245E-05
U1	144.00000		1541363.8	31.884728	166105.25	0.9036066	17.382145	1966.4421	8429.8872
U2	3504.6729		0.9731641	-0.4634934E-01	-0.1983702E 08	-750.63630	0.2187039E-01	0	2518.3734
U3	0		-0	89.341623	0.2107600E 08	894.63630	2042321.6	178.07620	8044.9236
U1	394.00000		665195.59	22.830292	578776.25	0.4977900	5.5167568	2898.4415	18265.592
U2	3504.6729		2.2549758	-0.2312308E-01	-0.1367041E 08	-938.58022	0.4847643E-01	8.2840270	1755.9968
U3	16		-0	89.341623	0.2148867E 08	1332.5802	0.1084321E 08	174.38141	18180.988
U1	394.00000		566282.04	22.830292	578776.25	0.4977900	5.5167568	2898.4415	18265.592
U2	294.11765		0.4414761	-0.2312308E-01	-0.1367041E 08	-938.58022	0.4847643E-01	8.2840270	1755.9968
U3	16		-0.1633231E 09	89.341623	0.2148857E 08	1332.5802	0.1084321E 08	174.38141	18180.988
U1	872.02884		425685.32	10.830313	750975.50	0.1726335E-03	-0.9653395E-04	5338.8524	25492.405
U2	294.11765		0.5872883	-0.2821767E-01	747312.25	888.28274	0.6743069E-01	35.552577	-0.4295050E-01
U3	29		-0	89.341623	0.2166087E 08	-16.253904	0.2166094E 08	-25.885881	25492.405
U1	872.02884		221168.00	10.830313	750975.50	0.1726335E-03	-0.9653395E-04	5338.8524	25492.405
U2	294.11765		0.5872883	-0.2821767E-01	747312.25	888.28274	0.6743069E-01	35.552577	-0.4295050E-01
U3	29		-0	89.341623	0.2166087E 08	-16.253904	0.2166094E 08	-25.885881	25492.405
U4	144.00000		0	4090909.0	0	367727.27	0	0.9175247	0
U5	250.00000		0	876168.22	0	98913.551	0	0.8985587	0
U6	478.02884		0	276887.54	0	68226.504	0	0.8023074	0

APPENDIX H

PROGRAM LISTING

```
      SUBROUTINE AERO
C
C          SUBROUTINE AERO COMPUTES MACH NUMBER, DYNAMIC PRESSURE, AND
C     DRAG ACCELERATIONS.  ACCELERATION VECTORS ARE REFERENCED TO THE
C     RELATIVE WIND VELOCITY.
C          THE DRAG COEFFICIENT IS ASSUMED TO BE A FUNCTION OF MACH
C     NUMBER AND A TABLE OF CD VS MACH NUMBER IS ASSUMED AS A FITTED
C     QUADRATIC EQUATION IN THE COEFFICIENT ARRAY.
C          THE CONSTANT IN THE Q EQUATION IS 0.5*1.4.  THE RATIO OF
C     SPECIFIC HEATS OF AIR IS ASSUMED TO BE A CONSTANT = 1.4.
C          PA HAS UNITS OF LB/FT/FT AND VELSD FT/SEC.
C
      COMMON /CSTAR/ CMA(1000),CMB(1000)
C
      DIMENSION DRAG  (3        ),RB    (5        ),VATM  (5        )
      DIMENSION VX    (5        )
      EQUIVALENCE (AREA  ,CMA(809)),(CD    ,CMA(812)),(DRAG  ,CMA(777))
      EQUIVALENCE (G     ,CMA(716)),(PA    ,CMA(806)),(Q     ,CMA(807))
      EQUIVALENCE (QVAL  ,CMA(808)),(RB    ,CMA(754)),(REVOLV,CMA(799))
      EQUIVALENCE (VATM  ,CMA(764)),(VELSD ,CMA(810)),(VMACH ,CMA(811))
      EQUIVALENCE (VX    ,CMA(759)),(WEIGHT,CMA(402))
C
C     COMPUTE RELATIVE VELOCITY
      VATM(1)=VX(1)+REVOLV*RB(2)
      VATM(2)=VX(2)-REVOLV*RB(1)
      VATM(3)=VX(3)
      VATM(4)=DOT(VATM,VATM)
      VATM(5)=SQRT(VATM(4))
C
C     COMPUTE MACH NUMBER
      VMACH=VATM(5)/VELSD
C
C
C     COMPUTE DYNAMIC PRESSURE
      Q=VMACH**2*PA*0.7
C
      QVAL=Q*AREA/WEIGHT*G
C
C     FIND DRAG COEFFICIENT
      CD=QUAD(VMACH,1)
C
C     COMPUTE DRAG ACCELERATIONS
      IF(VATM(5).EQ.0.0) RETURN
      DO 1 K=1,3
    1 DRAG(K)=-CD*QVAL*VATM(K)/VATM(5)
C
      RETURN
      END
```

```
      SUBROUTINE ATMOS
C
C          SUBROUTINE ATMOS SUPPLIES ATMOSPHERIC PRESSURE AND VELOCITY
C     OF SOUND AS A FUNCTION OF ALTITUDE.
C
      COMMON /SCHMOO/ A,AL,HARDC,PARDC,DENARD,TARDC,YPAT,ALTPAT,
     1GMOR,ONEBYR,P1,DENS1
      COMMON /CSTAR/ CMA(1000),CMB(1000)
      COMMON /ATABLE/ CME(8000)
C
      DIMENSION A      (6,4,3  ),AL     (10      ),DENARD(10      )
      DIMENSION HARDC  (10     ),PARDC  (10      ),POWER (3       )
      DIMENSION RB     (5      ),TARDC  (10      ),YPAT  (4       )
      EQUIVALENCE (ALT    ,CMA(805)),(APR    ,CME(004)),(B      ,CMA(814))
      EQUIVALENCE (CONM   ,CMA(717)),(CPA    ,CMA(719)),(OBLATN,CME(002))
      EQUIVALENCE (PA     ,CMA(806)),(R      ,CMA(758)),(RB     ,CMA(754))
      EQUIVALENCE (RBSQ   ,CMA(757)),(VELSD  ,CMA(810))
C
      DATA AMOL1/28.966/
      DATA AMOR/.34788119E-2/
C
C
C     DETERMINE IF OBLATE SPHEROIDAL OR SPHERICAL EARTH IS MODEL
      IF(OBLATN.NE.0.0) GO TO 1
C
      ALT=R-APR
      GO TO 2
    1 ALT=R-APR*B/ SQRT(B**2+(APR**2-B**2)*RB(3)**2/RBSQ)
C     CONVERT TO METERS FOR INTERNAL COMPUTATIONS
    2 ALTM=ALT/CONM
C     BELOW ALTPAT....USE PATRICK AFB MODEL FIT
C     ABOVE ALTPAT....USE ARDC 1959 MODEL FIT
      IF(ALTM-ALTPAT.GT.0.0) GO TO 6
      N = 1
C     DETERMINE ALTTABLE INTERVAL,YPAT.  INCREMENT N UNTIL LEVEL IS
C     FOUND.
    3 IF(ALTM.LT.YPAT(N)) GO TO 4
      N = N + 1
      GO TO 3
C     POLYNOMIAL IN ALTITUDE DETERMINED
    4 DO 5 J=1,3
    5 POWER(J) = ((((A(6,N,J)*ALTM + A(5,N,J))*ALTM + A(4,N,J))*ALTM +
     1 A(3,N,J))*ALTM + A(2,N,J))*ALTM + A(1,N,J)
      PRESS = P1*EXP(POWER(1))
      DENSE = DENS1*EXP(POWER(2))
      TEMP = POWER(3)
      GO TO 16
C     GEOPOTENTIAL ALTITUDE COMPUTED
    6 HALT = 6339260.0*ALTM/(ALTM + 6344520.0)
      N = 1
C     DETERMINE ALTITUDE TABLE INTERVAL, HARDC.  INCREMENT N UNTIL
C     LEVEL IS FOUND
    7 IF(HALT.LT.HARDC(N+1)) GO TO 8
      N = N +1
      GO TO 7
C     IF AL (TEMPERATURE GRADIENT) IS NOT CONSTANT (INDICATED BY
```

```
C     ZERO), COMPUTE PARAMETERS AS FUNCTIONS OF MOLECULAR WEIGHT        ZATM0057
C     RATIO.                                                            ZATM0058
    8 IF(AL(N).EQ.0.0) GO TO 9                                          ZATM0059
C     IF AL IS CONSTANT (NON-ZERO) COMPUTE WITH SIMPLIFIED              ZATM0060
C     RELATIONSHIP.                                                     ZATM0061
      TEMP = TARDC(N) + AL(N)*(HALT-HARDC(N))                           ZATM0062
      AMULT = TARDC(N)/TEMP                                             ZATM0063
      PRESS = PARDC(N)*AMULT**(GMOR/AL(N))                              ZATM0064
      GO TO 15                                                          ZATM0065
C     CONVERT TO KILOMETERS, FIND REGION BY HALTKM IN (0,90),           ZATM0066
C     (90,180), OR (180,INF).                                           ZATM0067
    9 HALTKM = HALT/1000.0                                              ZATM0068
      K = 1                                                             ZATM0069
      IF(HALTKM.GT.90.0) K = 2                                          ZATM0070
      IF(HALTKM.GT.180.0) K = 3                                         ZATM0071
      GO TO (10,11,12),K                                                ZATM0072
   10 TEMP = TARDC(N)                                                   ZATM0073
      GO TO 14                                                          ZATM0074
   11 AMOLE = 22.0 - 5.04483574*ATAN2(HALTKM-220.0,225.0)               ZATM0075
      GO TO 13                                                          ZATM0076
   12 AMOLE = 27.106 - 7.9356971*ATAN2(HALTKM-180.0,140.0)              ZATM0077
   13 TEMP = TARDC(N)*AMOL1/AMOLE                                       ZATM0078
   14 AMULT = EXP(-GMOR*(HALT-HARDC(N))/TARDC(N))                       ZATM0079
      PRESS = PARDC(N)*AMULT                                            ZATM0080
C     DENSITY, VELSD, PA COMPUTED FOR PROGRAM USE.                      ZATM0081
C     VELSD = (FPS), PA = (PSI).                                        ZATM0082
   15 DENSE =AMOR*PRESS/TEMP                                            ZATM0083
   16 VELSD = SQRT(1.40*PRESS/DENSE)*CONM                               ZATM0084
      PA = CPA*PRESS                                                    ZATM0085
      RETURN                                                            ZATM0086
      END                                                               ZATM0087
```

```
      FUNCTION ARCTAN (Y,X)                                             ZARC0001
                                                                        ZARC0002
C                                                                       ZARC0003
C          THE FORTRAN IV LIBRARY ATAN(+ OR - Z=TAN(THETA))USES A SINGLEZARC0004
C     ARGUMENT WITH ITS SIGN TO GIVE THETA IN THE FIRST (+Z) OR FOURTH  ZARC0005
C     (-Z) QUADRANT.  THE ARCTAN FUNCTION MAY BE USED IF + OR - Z IS    ZARC0006
C     DERIVED FROM A FRACTION SO THAT ARCTAN (Y/X) = TAN-1              ZARC0007
C     ((+OR-Y=SIN(THETA))/(+OR-X=COS(THETA))).  THUS THE ARCTAN(Y/X)    ZARC0008
C     GIVES THETA IN ITS PROPER QUADRANT FROM -180 DEGREES TO +180      ZARC0009
C     DEGREES.                                                          ZARC0010
C                                                                       ZARC0011
      IF (X) 2,1,2                                                      ZARC0012
    1 ARCTAN=SIGN(1.57079632,Y)                                         ZARC0013
      GO TO 4                                                           ZARC0014
    2 ARCTAN=ATAN(Y/X)                                                  ZARC0015
      IF (X) 3,1,4                                                      ZARC0016
    3 ARCTAN=ARCTAN+SIGN(3.14159265,Y)                                  ZARC0017
    4 RETURN                                                            ZARC0018
      END                                                               ZARC0019
```

```
      SUBROUTINE BOOST(VAR,NVAR,NKICKS)
C
C          SUBROUTINE BOOST PROVIDES A TWO-DIMENSIONAL SECOND ORDER
C     CURVE FIT OF BOOSTER BURNOUT CONDITIONS.  THE INTERPOLATION
C     EQUATION IS
C                         Z = A + BX + CY + DXY + EXX + FYY
C     THE PARTIAL DERIVATIVES OF THE ALTITUDE, RADIAL VELOCITY,
C     AND ANGULAR VELOCITY WITH RESPECT TO KICK ANGLE AND BOOSTER
C     BURNING TIME ARE OBTAINED BY DIFFERENTIATING THE INTERPOLATION
C     EQUATION.
C          THE COEFFICIENTS ARE CALCULATED BY DETERMINANTS.
C
      COMMON /CSTAR/ CMA(1000),CMB(1000)
      COMMON /ATABLE/ CME(8000)
      DIMENSION VAR(NVAR,NKICKS)
      DIMENSION A      (6       ),APTMAX(3       ),B      (6   ,6   )
      DIMENSION CON    (6   ,6  ),I      (6       ),ITB    (3        )
      DIMENSION ITK    (3       ),KS     (3       ),PAR    (6   ,2   )
      DIMENSION STB    (6       ),V      (6       )
      EQUIVALENCE (APTMAX,CMB(126)),(DELTB ,CME(016)),(DELTK ,CME(199))
      EQUIVALENCE (ITB   ,CMB(123)),(ITK   ,CMB(120)),(PAR   ,CMB(142))
      EQUIVALENCE (ROA   ,CMB(062)),(TBURN ,CMB(065)),(TMINST,CME(019))
      EQUIVALENCE (V     ,CMB(154))
      DATA (KS(I),I=1,3)/1,4,6/
C
C     SET UP DETERMINANT FOR COMPUTING D.  NOTE THAT THE KICK ANGLE
C     TERMS ARE NORMALIZED TO THE DESIRED KICK ANGLE (SEE MAINA) AND
C     THE BURNOUT TIME TERMS TO THE DESIRED BURNING TIME
      DO 1 J = 1,3
      K = KS(J)
      B(K,2)=APTMAX(J)/DELTK
    1 B(J,3)=(TMINST+FLOAT(ITB(J)-1)*DELTB-TBURN)/DELTB
      B(2,2)=B(1,2)
      B(3,2)=B(1,2)
      B(5,2)=B(4,2)
      B(4,3)=B(1,3)
      B(6,3)=B(1,3)
      B(5,3)=B(2,3)
      KA=0
      DO 2 J = 1,3
      JA=4-J
      DO 2 K=1,JA
      KA=KA+1
      I(KA)=25*(ITK(J)-1)+ITB(K)
      B(KA,1)=1.0
      B(KA,4)=B(KA,2)*B(KA,3)
      B(KA,5)=B(KA,2)**2
    2 B(KA,6)=B(KA,3)**2
C
C     CALL DETERM TO COMPUTE D
      CALL DETERM(B,D,6,6)
C
C     COMPUTE COEFFICIENTS (CON) OF INTERPOLATION EQUATION.
      J1=I(1)
      DO 5 K = 1,NVAR
      DO 3 M = 1,6
```

```
      J2=I(M)
    3 A(M)=VAR(K,J2)-VAR(K,J1)
      DO 5 L = 1,6
      DO 4 J = 1,6
      STB(J)=B(J,L)
    4 B(J,L)=A(J)
      CALL DETERM(B,CON(L,K),6,6)
      DO 5 J=1,6
    5 B(J,L)=STB(J)
C
      DO 6 K = 1,NVAR
C
C     V(1) ALTITUDE
C     V(2) RADIAL VELOCITY
C     V(3) HORIZONTAL VELOCITY
      V(K)=CON(1,K)/D+VAR(K,J1)
C
C     PAR CONTAINS DERIVATIVES
C     PAR(K,1) WITH RESPECT TO BURNING TIME.
C     PAR(K,2) WITH RESPECT TO KICK ANGLE.
      PAR(K,1)=CON(3,K)/D/DELTB
    6 PAR(K,2)=CON(2,K)/D/DELTK
      RAD = ROA+V(1)
      DO 7 K = 1,2
    7 PAR(3,K)=PAR(3,K)/RAD-V(3)/RAD**2*PAR(1,K)
C
      RETURN
      END
```

```
      SUBROUTINE CHECK                                                  ZCHE0001
C                                                                       ZCHE0002
C          SUBROUTINE CHECK COMPARES THE PROPELLANT LOADINGS OF THE     ZCHE0003
C     OPTIMIZED CASE WITH PRELOADED VALUES.  IF A PROPELLANT LOADING    ZCHE0004
C     (FUEL) EXCEEDS ITS LIMIT (WPMAX), THE BURNING TIME IS SHORTENED TOZCHE0005
C     THE TIME  SUCH THAT EXACTLY THE SPECIFIED AMOUNT OF PROPELLANT IS ZCHE0006
C     UTILIZED AND PHASE TIME IS FIXED.                                 ZCHE0007
C                                                                       ZCHE0008
      COMMON /CSTAR/ CMA(1000),CMB(1000)                                ZCHE0009
      DIMENSION FLOMX (6       ),FUEL  (6       ),JEND  (20      )      ZCHE0010
      DIMENSION NOPT  (6       ),TB    (6       ),TK    (6       )      ZCHE0011
      DIMENSION WPMAX (6       )                                        ZCHE0012
      EQUIVALENCE (FLOMX ,CMA(837)),(FUEL  ,CMA(871)),(FUELDV,CMA(878)) ZCHE0013
      EQUIVALENCE (JDATA ,CMA(925)),(LAST  ,CMA(890)),(NOPT  ,CMA(819)) ZCHE0014
      EQUIVALENCE (TB    ,CMA(825)),(WPMAX ,CMA(855))                   ZCHE0015
      DATA (TK(I),I=1,6)/6H FIRST,6HSECOND,6H THIRD,6HFOURTH,6H FIFTH,6HZCHE0016
     1 SIXTH/                                                           ZCHE0017
      DO 2 J = 1,LAST                                                   ZCHE0018
C                                                                       ZCHE0019
C     CHECK FOR EXCESS PROPELLANT UTILIZATION BY EACH PHASE             ZCHE0020
      IF(WPMAX(J).EQ.0.0.OR.FUEL(J).LE.WPMAX(J)) GO TO 2                ZCHE0021
C                                                                       ZCHE0022
C     IF JDATA EQUALS THREE, THE CASE WITH THE EXCESS PROPELLANT        ZCHE0023
C     HAS BEEN PRINTED.                                                 ZCHE0024
      IF(JDATA.NE.3) GO TO 1                                            ZCHE0025
C                                                                       ZCHE0026
C     FIX BURNING TIME OF OFFENDING PHASE                               ZCHE0027
      TB(J)=WPMAX(J)/FLOMX(J)                                           ZCHE0028
C                                                                       ZCHE0029
      WRITE (6,3 ) TK(J),TB(J)                                          ZCHE0030
C                                                                       ZCHE0031
C     SET NOPT TO ZERO FOR OFFENDING PHASE                              ZCHE0032
    1 NOPT(J)=0                                                         ZCHE0033
C                                                                       ZCHE0034
    2 CONTINUE                                                          ZCHE0035
    3 FORMAT (1H0,1X,1A6,37H STAGE BURNING TIME HAS BEEN FIXED AT,      ZCHE0036
     11X,G14.7,9H SECONDS.)                                             ZCHE0037
      RETURN                                                            ZCHE0038
      END                                                               ZCHE0039
```

```
      SUBROUTINE COAST                                                  ZCOA0001
C                                                                       ZCOA0002
C          SUBROUTINE COAST TERMINATES A PHASE WHEN SOME DESIGNATED      ZCOA0003
C     PARAMETER REACHES A DESIRED VALUE.  TO ENABLE THE USER TO          ZCOA0004
C     FAMILIARIZE HIMSELF WITH THE ORIGIN AND UTILITY OF THESE           ZCOA0005
C     TERMINATING EQUATIONS, THE EQUATION NUMBERS FROM PAYLOAD           ZCOA0006
C     OPTIMIZATION OF MULTISTAGE LAUNCH VEHICLES (NASA TN-3121)          ZCOA0007
C     BY THE AUTHORS IS PROVIDED.  CAPPAD IS THE DESIRED VALUE           ZCOA0008
C     WHICH CAPPIN MUST EQUAL FOR STAGE TERMINATION.  IT MAY BE          ZCOA0009
C     SEEN THAT IN THE AFOREMENTIONED REPORT, THE CAPPAD S               ZCOA0010
C     APPEAR ON THE RIGHT SIDE OF THE EQUAL SIGNS AND THE CAPPIN S       ZCOA0011
C     ON THE LEFT SIDE.                                                  ZCOA0012
C          THIS ROUTINE CHECKS AT EACH INTEGRATION STEP FOR A CHANGE     ZCOA0013
C     IN SIGN IN CAPPA WHERE CAPPA = CAPPIN - CAPPAD AND WHEN CAPPA      ZCOA0014
C     CHANGES SIGN, COAST CONTROLS THE INTEGRATION STEP SIZE TO          ZCOA0015
C     ZERO CAPPA.                                                        ZCOA0016
C                                                                       ZCOA0017
      COMMON /CSTAR/ CMA(1000),CMB(1000)                                ZCOA0018
      COMMON /RUNG/RUN(125)                                             ZCOA0019
      DIMENSION FLOMX (6      ),FUEL  (6      ),HARD  (6      )         ZCOA0020
      DIMENSION IDATA (6   ,5  ),JCOAST(6      ),NOPT  (6      )         ZCOA0021
      DIMENSION PROP  (6      ),S     (6   ,2  ),TB    (6      )         ZCOA0022
      DIMENSION THRUST(6      ),TS    (6      )                         ZCOA0023
      EQUIVALENCE (CAPPA ,CMA(891)),(DELT  ,CMA(701)),(DELTAV,CMA(861)) ZCOA0024
      EQUIVALENCE (ENERGY,CMA(892)),(ERRMXK,CMB(054)),(FLOMX ,CMA(837)) ZCOA0025
      EQUIVALENCE (FLOW  ,CMA(877)),(FM    ,CMA(715)),(FUEL  ,CMA(871)) ZCOA0026
      EQUIVALENCE (G     ,CMA(716)),(HARD  ,CMA(843)),(H2    ,RUN(106)) ZCOA0027
      EQUIVALENCE (I     ,RUN(107)),(IDATA ,CMB(086)),(JCOAST,CMB(130)) ZCOA0028
      EQUIVALENCE (JCOST ,CMB(129)),(LAST  ,CMA(890)),(MASH  ,CMB(064)) ZCOA0029
      EQUIVALENCE (MODOUT,CMA(714)),(NCUTE ,CMA(893)),(NOPT  ,CMA(819)) ZCOA0030
      EQUIVALENCE (NOPTA ,CMB(070)),(NSTAGE,CMA(710)),(OMEGA ,CMA(405)) ZCOA0031
      EQUIVALENCE (PROP  ,CMA(849)),(R     ,CMA(402)),(RMASS ,CMA(401)) ZCOA0032
      EQUIVALENCE (S     ,CMB(074)),(TB    ,CMA(825)),(THRUST,CMA(831)) ZCOA0033
      EQUIVALENCE (TIME  ,CMA(409)),(TS    ,CMA(932)),(U     ,CMA(404)) ZCOA0034
      EQUIVALENCE (V     ,CMA(889)),(VELEX ,CMA(870)),(ZLAM1 ,CMA(406)) ZCOA0035
      EQUIVALENCE (ZLAM2 ,CMA(407)),(ZLAM3 ,CMA(408)),(ZLAM4 ,CMA(884)) ZCOA0036
C                                                                       ZCOA0037
C     DETERMINE PROPER EQUATION FOR TERMINATING PHASE USING INFORMATION ZCOA0038
C     COLLECTED IN XOLOAD.  EVALUATE CONSTANT TERMS IN THE PROPER       ZCOA0039
C     EQUATION.                                                         ZCOA0040
      GO TO (28,11,11,12,14,1,15,16,13),JCOST                           ZCOA0041
    1 KA=JCOAST(NSTAGE)                                                 ZCOA0042
      K = 0                                                             ZCOA0043
      IF(KA.NE.0) GO TO 3                                               ZCOA0044
      IF(NSTAGE.EQ.LAST.AND.NCUTE.EQ.1) GO TO  2                        ZCOA0045
      JCOST=1                                                           ZCOA0046
      RETURN                                                            ZCOA0047
C                                                                       ZCOA0048
C     SET CAPPAD EQUAL TO THE DESIRED ENERGY                            ZCOA0049
    2 CAPPAD=ENERGY                                                     ZCOA0050
      JCOST=7                                                           ZCOA0051
      TS(NSTAGE)=TS(NSTAGE-1)+10000.0                                   ZCOA0052
C                                                                       ZCOA0053
      GO TO 10                                                          ZCOA0054
    3 GO TO (4,6),KA                                                    ZCOA0055
    4 IF(NSTAGE.EQ.LAST) GO TO 5                                        ZCOA0056
```

```
C
C     RIGHT HAND SIDE OF EQUATION 47
      CAPPAD=(1.0+PROP(NSTAGE))/PROP(NSTAGE)*(S(NSTAGE,1)-S(NOPTA,2)/
     1(1.0+PROP(NOPTA))-SADDA(NOPTA+1,NSTAGE-1))
      JCOST=2
C
      GO TO 10
C
C     RIGHT HAND SIDE OF EQUATION 50
    5 CAPPAD=-((1.0+PROP(LAST))*EXP(-DELTAV/VELEX))/PROP(LAST)*(S(NOPTA,
     12 )/(1.0+PROP(NOPTA))-S(LAST,1)+SADDA(NOPTA+1,LAST-1))
      JCOST=3
C
      GO TO 10
    6 IF(IDATA(NSTAGE+1,5).EQ.1) GOTO 9
      IF(NSTAGE.GT.NOPTA+1) GO TO 7
C
C     RIGHT HAND SIDE OF EQUATION 48B
      CAPPAD=0.0
      JCOST=8
C
      GO TO 10
C
C     RIGHT HAND SIDE OF EQUATION 48A
    7 CAPPAD=SADDB(1,NOPTA+1,NSTAGE)-SADDB(2,NOPTA+1,NSTAGE-1)
     1-S(NOPTA,2)/(1.0+PROP(NOPTA))
C
      IF(FLOMX(NSTAGE).EQ.0.0) GO TO 8
      JCOST=4
      GO TO 10
    8 JCOST=9
      GO TO 10
C
C     RIGHT HAND SIDE OF EQUATION 41C
    9 CAPPAD=0.0
C
      JCOST=5
C
   10 GO TO (11,11,11,12,14,15,15,16,13),JCOST
C
C     LEFT HAND SIDE OF EQUATION 50
   11 CAPPIN=SSTAGE(THRUST(NSTAGE),FLOW,RMASS)
C
      GO TO 17
C
C     LEFT HAND SIDE OF EQUATION 48A
   12 CAPPIN=((FM/R**2-OMEGA**2*R)*ZLAM1+2.0*U*OMEGA*ZLAM2-U*ZLAM3
     1-OMEGA*ZLAM4)/(1.0/FLOMX(NSTAGE)-1.0/FLOMX(NSTAGE+1))-G*SQRT
     1(ZLAM1**2+ZLAM2**2)*(THRUST(NSTAGE)/RMASS/FLOMX(NSTAGE)+THRUST
     1(NSTAGE+1)/(RMASS-HARD(NSTAGE)-PROP(NSTAGE)*(TIME-TS(NSTAGE-1))*
     1FLOW)/FLOMX(NSTAGE+1))
C
      GO TO 17
C
C     LEFT HAND SIDE OF EQUATION 48A WHEN FLOW = 0
   13 CAPPIN=-SSTAGE(THRUST(NSTAGE+1),FLOMX(NSTAGE+1),RMASS-HARD(NSTAGE
     1))
C
      GO TO 17
C
```

```
C     LEFT HAND SIDE OF EQUATION 41C
   14 CAPPIN=(FM/R**2-OMEGA**2*R)*ZLAM1+2.0*U*OMEGA*ZLAM2-U*ZLAM3-OMEGA
     1*ZLAM4
C
      GO TO 17
C
C     CALCULATE ENERGY
   15 CAPPIN=V**2/2.0-FM/R
C
      GO TO 17
C
C     LEFT HAND SIDE OF EQUATION 48B
   16 CAPPIN=SSTAGE(THRUST(NSTAGE+1),FLOMX(NSTAGE+1),RMASS-HARD(NSTAGE)-
     1PROP(NSTAGE)*(TIME-TS(NSTAGE-1))*FLOW)-SSTAGE(THRUST(NSTAGE),
     1FLOW,RMASS)/(1.0+PROP(NOPTA))
C
C     COMPUTE CAPPA AND DCAPPA
   17 CAPPA=CAPPIN-CAPPAD
      DCAPPA=(CAPPA-CAPPAI)/DELT
      IF(K.NE.0) GOTO 18
      CAPPAI=CAPPA
      K=1
   18 GO TO (19,20,28,28),I
C
C     CHECK FOR CAPPA SIGN CHANGE
   19 IF(CAPPA*CAPPAI.LT.0.0) GOTO 24
C
C     CHECK FOR DIVERGING CAPPA
      IF(ABS(CAPPA).LE.ABS(CAPPAI)) GO TO 27
C
C     DELT = 0  TERMINATES INTEGRATION
      DELT=0.0
C
C     I = 2 CAUSES CONTROL TO BYPASS STEP-SIZE CONTROL IN STEP
      I = 2
C
C     MASH = 1 IS A FLAG TO MAIN THAT THE PROPOSED CASE IS IMPOSSIBLE.
      MASH=1
      RETURN
C
C
C     CHECK FOR CAPPA WITHIN CONVERGENCE TOLERANCE
   20 DELCAP=DELT3*DCAPPA
      IF (DELCAP.EQ.0.0) GO TO 21
      PERRK=CAPPA/DELCAP
      IF(ABS(PERRK).GE.ERRMXK) GO TO 25
C
C
C     CAPPA WITHIN TOLERANCE.  SET UP FOR CONTINUING THE PROBLEM.
   21 K=0
C
C     I EQUAL TO FOUR CAUSES CONTROL OF STEP-SIZE TO REVERT TO STEP
      I=4
C
C     SEE RUNGEK FOR DESCRIPTION OF H2
      H2=TIME-TSTO
C
C     RESET DELT
      DELT=DELT3
C
```

```
C     RESET JCOST FOR NEXT PHASE
      JCOST=6
C     SET TS TO ENABLE STEP TO INCREMENT NSTAGE
      TS(NSTAGE)=TIME
C
C     COMPUTE PHASE BURNING TIME
      TB(NSTAGE)=TS(NSTAGE)-TS(NSTAGE-1)
C
      JA=NSTAGE+1
C     COMPUTE PROPELLANT LOADING FOR PHASE
      FUEL(NSTAGE)=TB(NSTAGE)*FLOW
      IF(NSTAGE.EQ.LAST) RETURN
      DO 22 J = JA,LAST
C
C     JCOAST(J) NOT EQUAL TO ZERO IMPLIES THAT ANOTHER PHASE WILL BE
C     TERMINATED BY COAST.
      IF(JCOAST(J).NE.0) GO TO 23
C
C     COMPUTE PROPELLANT LOADINGS FOR REMAINING PHASES
      FUEL(J)=TB(J)*FLOMX(J)
C
C     SET TS S FOR REMAINING PHASES
   22 TS(J)=TS(J-1)+TB(J)
      RETURN
C
C     SET TS VERY LARGE FOR NEXT PHASE TERMINATED BY COAST
   23 TS(J)=TS(J-1)+10000.0
      RETURN
C     I EQUAL TO TWO CAUSES STEP TO RELINQUISH CONTROL OF STEP-SIZE
   24 I = 2
C
C     STORE DATA FOR COMPUTING H2 WHEN CAPPA IS ZEROED.
      TSTO=TIME-H2
C
C     STORE DELT
      DELT3=H2
C
C     COMPUTE DELT TO ZERO CAPPA
   25 DELT=CAPPA/(CAPPAI-CAPPA)*DELT
C
   26 IF(ABS(DELT).LT. ABS(DELT3)) GO TO 27
      DELT=DELT/2.0
      GO TO 27
C
C     STORE CAPPA FOR LAST INTEGRATION STEP.
   27 CAPPAI=CAPPA
C
   28 RETURN
      END
```

```
      SUBROUTINE COEFNT
C
C          THIS SUBROUTINE COMPUTES COEFFICIENTS OF SECOND ORDER
C     POLYNOMIALS USED TO FIT VARIOUS FUNCTION CURVES SUCH AS CD VS
C     MACH NUMBER.  DATA IS INPUT AS A TWO COLUMN ARRAY.  THE FIRST
C     COLUMN IS THE INDEPENDENT VARIABLE (MACH) AND THE SECOND
C     COLUMN THE CORRESPONDING DEPENDENT VALUE (CD).  ALL SETS OF
C     DATA POINTS ARE LOADED SEQUENTIALLY AND IDENTIFIED BY NSETS(K)
C     WHICH IS THE NUMBER OF SETS OF THREE DATA POINTS NEEDED
C     TO SOLVE AN EQUATION OF THE FORM CD = A + BM + CMM.
C
C
C     NUMBER OF POINTS MUST BE ODD
C     ICC HAS POSITIONS OF BEGINNING OF ARRAY IN COEFN.
C     NSETS REFERS TO THE NUMBER OF SETS AND IT EQUALS
C               (NUMBER OF POINTS -ONE)/2.0.
C     NSAVE = POSITIONS IN COEFN FOR THE SETS.
C     VARIND AND VARDEP ARE LOADED CONSECUTIVELY.
C     STORAGE IN COEFN MUST ALLOT FOUR*NSETS(N)+1 FOR EACH SET.
C
C
      COMMON /CSTAR/ CMA(1000),CMB(1000)
C
      DIMENSION COEFN (500     ),JEND  (20      ),NSAVE (20      )
      DIMENSION NSETS (20      ),V     (100,2  )
      EQUIVALENCE (COEFN ,CMB(501)),(NSAVE ,CMB(241)),(NSETS ,CMB(221))
      EQUIVALENCE (V     ,CMB(301))
      DATA JS/0/
C
      J=1
      DO 2 K=1,20
      I=NSETS(K)
      IF(I.EQ.0) GO TO 3
      N=NSAVE(K)
      DO 1 L=1,I
      S=V(J+1,1)*V(J+2,1)**2-V(J+2,1)*V(J+1,1)**2
      T=V(J+2,1)*V(J,1)**2-V(J,1)*V(J+2,1)**2
      U=V(J,1)*V(J+1,1)**2-V(J+1,1)*V(J,1)**2
      D=S+T+U
      COEFN(N)=V(J,1)
      COEFN(N+1)=(V(J,2)*S+V(J+1,2)*T+V(J+2,2)*U)/D
      COEFN(N+2)=(V(J,2)*(V(J+1,1)**2-V(J+2,1)**2)+V(J+1,2)*(V(J+2,1)**
     12-V(J,1)**2)+V(J+2,2)*(V(J,1)**2-V(J+1,1)**2))/D
      COEFN(N+3)=(V(J,2)*(V(J+2,1)-V(J+1,1))+V(J+1,2)*(V(J,1)-V(J+2,1))+
     1V(J+2,2)*(V(J+1,1)-V(J,1)))/D
      J=J+2
    1 N=N+4
      COEFN(N)=V(J,1)*10.0
      J=J+1
    2 JEND(K)=N
      JE = N-1
    3 DO 4 K=1,20
      IF(NSETS(K).EQ.0) RETURN
      JS=NSAVE(K)
      JE=JEND(K)
      NSETS(K)=0
```

```
      WRITE (6,6) JS,JE                                                 ZCOE0057
      WRITE (6,7 )(COEFN(M),M=JS,JE)                                    ZCOE0058
    4 WRITE(6,5 )(COEFN(M),M=JS,JE)                                     ZCOE0059
    5 FORMAT(2H*0G14.7,1H,,G14.7,1H,,G14.7,1H,,G14.7,3H,$$)             ZCOE0060
    6 FORMAT (13HOCOEFFICIENTSI4,1H,I3)                                 ZCOE0061
    7 FORMAT(6G20.8)                                                    ZCOE0062
      RETURN                                                            ZCOE0063
      END                                                               ZCOE0064
```

```
      SUBROUTINE CONEVL (FLOW,N)                                        ZCON0001
C                                                                       ZCON0002
C          SUBROUTINE CONEVL COMPUTES THE CONSTANT IN EQUATION          ZCON0003
C     11 WHEN THE THRUST IS EQUAL TO ZERO.  THE EQUATION APPEARS        ZCON0004
C     AGAIN IN 41C.  CONST(N,1) IS USED IN FINAL AS A FINAL             ZCON0005
C     CONDITION FOR AN OPTIMUM COAST.  CONST(N,2) IS THE SQUARE         ZCON0006
C     ROOT OF THE SUM OF THE SQUARES OF THE TERMS IN CONST(N,1)         ZCON0007
C     AND IS USED IN SCOMP.                                             ZCON0008
      COMMON /CSTAR/ CMA(1000),CMB(1000)                                ZCON0009
      DIMENSION CONST (5   ,2   )                                       ZCON0010
      EQUIVALENCE (CONST ,CMA(894)),(FM    ,CMA(715)),(G     ,CMA(716)) ZCON0011
      EQUIVALENCE (OMEGA ,CMA(405)),(R     ,CMA(402)),(U     ,CMA(404)) ZCON0012
      EQUIVALENCE (ZLAM1 ,CMA(406)),(ZLAM2 ,CMA(407)),(ZLAM3 ,CMA(408)) ZCON0013
      EQUIVALENCE (ZLAM4 ,CMA(884))                                     ZCON0014
      IF (FLOW.EQ.0.0) GO TO 1                                          ZCON0015
      CONST(N,1)=0.0                                                    ZCON0016
      CONST(N,2)=0.0                                                    ZCON0017
      GO TO 2                                                           ZCON0018
    1 CONST(N,1)= (FM/R**2-OMEGA**2*R)*ZLAM1+2.0*U*OMEGA*ZLAM2-U*       ZCON0019
     1ZLAM3-OMEGA*ZLAM4                                                 ZCON0020
      CONST(N,2)=SQRT (((FM/R**2-OMEGA**2*R)*ZLAM1)**2+(2.0*U*OMEGA*    ZCON0021
     1ZLAM2)**2+(U*ZLAM3)**2+(OMEGA*ZLAM4)**2)                          ZCON0022
    2 RETURN                                                            ZCON0023
      END                                                               ZCON0024
```

```
      SUBROUTINE CONVT(A,B,C)                                           ZCON0001
C                                                                       ZCON0002
C          CONVT TAKES THE VECTOR PRODUCT OF VECTORS A AND B AND PLACES ZCON0003
C     THE RESULT IN C.                                                  ZCON0004
C                                                                       ZCON0005
      DIMENSION A(5),B(5),C(5),IND(3)                                   ZCON0006
C                                                                       ZCON0007
      DATA(IND(I),I=1,3)/2,3,1/                                         ZCON0008
      DO 1 J1=1,3                                                       ZCON0009
      J2=IND(J1)                                                        ZCON0010
      J3=IND(J2)                                                        ZCON0011
    1 C(J3)=A(J1)*B(J2)-A(J2)*B(J1)                                     ZCON0012
      C(4)=C(1)*C(1)+C(2)*C(2)+C(3)*C(3)                                ZCON0013
      C(5)=SQRT(C(4))                                                   ZCON0014
      RETURN                                                            ZCON0015
      END                                                               ZCON0016
```

```
      SUBROUTINE DAMODE(N,OMIT)
C
C          SUBROUTINE DAMODE CONVERTS THE MODE INFORMATION WHICH
C     ENTERS THROUGH THE INPUT ROUTINE INTO MODOUT AND MODS WHICH
C     ARE USED IN STEP (SEE STEP).  ALSO TMIN AND STEPS ACQUIRE
C     PROPER VALUES FROM INPUT DATA.  THE INITIAL DELT IS COMPUTED
C     IN THIS ROUTINE.  THIS ROUTINE IS APPLICABLE TO THE UPPER
C     PHASES ONLY.
C
      COMMON /CSTAR/ CMA(1000),CMB(1000)
      DIMENSION DELMX (3      ),MODE   (3      ),STEP   (3      )
      DIMENSION TMIN1 (3      ),TS     (6      )
      EQUIVALENCE (DELMAX,CMA(702)),(DELMX ,CMB(163)),(DELT   ,CMA(701))
      EQUIVALENCE (DELTST,CMA(931)),(K     ,CMB(182)),(MODE   ,CMB(160))
      EQUIVALENCE (MODOUT,CMA(714)),(MODS  ,CMA(712)),(NPRINT,CMB(181))
      EQUIVALENCE (NSTAGE,CMA(710)),(STEP  ,CMB(166)),(STEPS ,CMA(704))
      EQUIVALENCE (TIME  ,CMA(609)),(TMIN  ,CMA(703)),(TMIN1 ,CMB(169))
      EQUIVALENCE (TS    ,CMA(932))
      INTEGER STEPS,STEP,OMIT
C
C     K IS USED IN STEP AS AN INDEX OF NOUT
      K = MIN0(N,2)
C
C
C     NPRINT NOT EQUAL TO ZERO CAUSES THE PERTURBATIONS TO BE
C     PRINTED.  OMIT ASSUMES VALUES FROM ONE TO SIX.
      IF(NPRINT.EQ.0.AND.OMIT.NE.1) GO TO 2
C
C
C     SET MODOUT
      MODOUT = MODE(N)/10
C
C
C     SET MODS
      MODS=MOD(MODE(N),10)
C
C
C     SET DELMAX
      DELMAX = DELMX(N)
C
C     SET MODOUT, TMIN, AND STEPS FOR MODE = 71,72,73,OR 74
      IF(MODOUT.NE.7) GO TO 1
      MODOUT=1
      TMIN=DELMAX-AMOD(TIME+DELMAX,DELMAX)
      IF(TMIN.EQ.0.0.OR.TMIN.LT.(.1)*DELMAX) TMIN=TMIN+DELMAX
      TMIN=TIME+TMIN
      STEPS=1000
      GO TO 3
C
C     SET STEPS
    1 STEPS = STEP(N)
C
C     SET TMIN
      TMIN = TMIN1(N)
C
      GO TO 3
```

```
C                                                                       ZDAM0057
C     MODOUT SET EQUAL TO SIX WHEN THE PERTURBATIONS ARE NOT PRINTED.   ZDAM0058
    2 MODOUT=6                                                          ZDAM0059
C                                                                       ZDAM0060
C     SET DELT                                                          ZDAM0061
    3 DELT=AMIN1(DELTST,TS(NSTAGE)-TIME)                                ZDAM0062
      IF(MODOUT.EQ.1.AND.DELT.GT.(TMIN-TIME)) DELT = (TMIN-TIME)        ZDAM0063
      IF((MODOUT.EQ.2.OR.MODOUT.EQ.3).AND.DELT.GT.DELMAX) DELT = DELMAX ZDAM0064
C                                                                       ZDAM0065
      RETURN                                                            ZDAM0066
      END                                                               ZDAM0067
```

```
C     DATATM                                                            ZGDA0001
      BLOCK DATA                                                        ZGDA0002
C                                                                       ZGDA0003
C          DATATM CONTAINS THE CONSTANTS ATMOS USES TO COMPUTE PRESSURE ZGDA0004
C     AND VELOCITY OF SOUND.                                            ZGDA0005
C                                                                       ZGDA0006
C     A            POLYNOMIAL COEFFICIENTS (PAFB FIT)                   ZGDA0007
C     AL           TEMPERATURE GRADIENTS AND/OR FLAGS (ARDC)            ZGDA0008
C     TARDC        TEMP. AND BASE OF INTERVAL (ARDC)                    ZGDA0009
C     HARDC        ALT. AT BASE OF INTERVAL (ARDC)                      ZGDA0010
C     DENARD       DENSITY AT BASE OF INTERVAL (ARDC)                   ZGDA0011
C     YPAT         ALT AT BASE OF INTERVAL  (PAFB)                      ZGDA0012
C     ALTPAT       SWITCH ALT PAFB TO ARDC                              ZGDA0013
C     GMOR         GRAV. CONST (GM) DIV BY GAS CONST (R)                ZGDA0014
C     ONEBYR       RATIO 1.0 AND R                                      ZGDA0015
C     P1           TEN NEWTONS/CM**2, REF PAFB PA                       ZGDA0016
C     DENS1        1000 NEWT/CM**3, REF PAFB DENSITY                    ZGDA0017
C     PARDC        PRESSURE AT BASE....                                 ZGDA0018
C                                                                       ZGDA0019
      COMMON /SCHMOO/ A,AL,HARDC,PARDC,DENARD,TARDC,YPAT,ALTPAT,        ZGDA0020
     1GMOR,ONEBYR,P1,DENS1                                              ZGDA0021
C                                                                       ZGDA0022
C      ATMOSPHERIC COEFFICIENTS USED BY GD/A IN SIMULATING              ZGDA0023
C     PATRICK REFERENCE ATMOSPHERE (BELOW 47.3904KM.) AND THE 1959      ZGDA0024
C     ARDC MODEL (ABOVE 47.3904).     REF. GDA63-0439-1  AC-4 HDBK.     ZGDA0025
C                                                                       ZGDA0026
      DIMENSION                                                         ZGDA0027
     1A(6,4,3), HARDC(10), AL(10), TARDC(10), DENARD(10), PARDC(10)     ZGDA0028
      DIMENSION  YPAT (4)                                               ZGDA0029
C                                                                       ZGDA0030
      DATA                                                              ZGDA0031
     4A/1.6871582E-2,-1.1425176E-4,-1.3612327E-9,7.3624145E-14,         ZGDA0032
     5-1.0800315E-17,3.3046432E-22,-7.9910777E-2,-8.1046438E-5,-5.552238ZGDA0033
     63E-9,3.1116969E-13,-1.6687827E-17,3.8319351E-22,9.8414277E-1,-2.69ZGDA0034
     776917E-4,8.5527541E-9,-3.9620263E-13,1.0146471E-17,-1.0264318E-22,ZGDA0035
     8-5.0773833E-1,-9.8961948E-5,-9.2758563E-11,-1.0270700E-13,3.172489ZGDA0036
     92E-18,-2.6699639E-23,                                             ZGDA0037
     11.3302117E-2,-8.8502064E-5,-4.2143056E-9,5.9517557E-13,-3.9744789EZGDA0038
     2-17,7.8771273E-22,1.2667122E-1,-1.3373147E-4,2.0667371E-9,2.339610ZGDA0039
     39E-13,-3.2562503E-17,7.9035209E-22,9.2751266E-1,-1.4349679E-4,-2.8ZGDA0040
     4271736E-9,4.7480092E-14,1.8863246E-18,-4.2702411E-23,8.3777777E-5,ZGDA0041
     5-1.5345007E-4,4.5202979E-9,-2.6563772E-13,5.5558369E-18,-3.8876587ZGDA0042
     6E-23,                                                             ZGDA0043
     72.9667877E2,-6.7731001E-3,8.4619805E-7,-1.7004049E-10,1.1451454E-1ZGDA0044
     84,-2.4898788E-19,2.6892151E2,4.3075352E-3,-8.9159672E-7,-2.8929791ZGDA0045
     99E-11,5.0724856E-15,-1.1490372E-19,3.7064557E2,-3.2858965E-2,2.0645ZGDA0046
     1636E-6,-4.3283944E-11,-5.7507242E-17,8.2924583E-21,5.8825814E2,-2.ZGDA0047
     27515685E-2,1.6593611E-7,3.0416388E-11,-8.0264768E-16,6.010202E-21/ZGDA0048
       DATA                                                             ZGDA0049
     1AL/0.0,-.0045,0.0,.0040,.0200,.0100,.0050,3*.0050/,               ZGDA0050
     2TARDC/282.66,282.66,165.66,165.66,225.66,1325.66,1425.66,1575.66, ZGDA0051
     3 2*0.0/,                                                          ZGDA0052
     4HARDC/47000.,53000.,79000.,90000.,105000.,160000.,170000.,200000.,ZGDA0053
     5 2.0E06,1.0E07/,                                                  ZGDA0054
     6DENARD/1.4851E-3,7.1944E-4,2.122E-5,2.194E-6,1.141E-7,9.522E-6,   ZGDA0055
     7 6.906E-6,3.153E-10,2*0.0/,                                       ZGDA0056
```

```
     1YPAT/10832.1,17853.3,28561.8,47390.4/,ALTPAT/47390.4/,              ZGDA0057
     2GMOR,ONEBYR/.34141626E-1,.1202734 E-3/,                             ZGDA0058
     5P1,DENS1/10197.162,0.1190934/,                                      ZGDA0059
     8PARDC/1.2283E01,5.9477,1.029E-1,1.066E-2,7.618E-4,3.691E-5,        ZGDA0060
     9 2.880E-5,1.454E-5,2*0.0/                                           ZGDA0061
      END                                                                 ZGDA0062
```

```
      SUBROUTINE GUESS                                                    ZGUE0001
C                                                                         ZGUE0002
C          SUBROUTINE GUESS IS A DUMMY ROUTINE CALLED AT THE              ZGUE0003
C     BEGINNING OF EACH PROBLEM AND WAS PLACED THERE FOR                  ZGUE0004
C     PREPARING GUESSES FOR THE INITIAL CONDITIONS BASED ON               ZGUE0005
C     PREVIOUSLY CONVERGED CASES.                                         ZGUE0006
C                                                                         ZGUE0007
      RETURN                                                              ZGUE0008
      END                                                                 ZGUE0009
```

```
C     DATAB                                                             ZDAT0001
C          DATAB IS A BLOCK DATA ROUTINE CONTAINING THE CONSTANTS FOR   ZDAT0002
C     THE PROGRAM.                                                      ZDAT0003
C                                                                       ZDAT0004
      BLOCK DATA                                                        ZDAT0005
C     DATAB                                                             ZDAT0006
      COMMON/ATABLE/CME(8000)                                           ZDAT0007
      COMMON /CSTAR/ CMA(1000),CMB(1000)                                ZDAT0008
      DIMENSION ANGLES(4      ),DELMX (3      ),DELMXB(2      )         ZDAT0009
      DIMENSION ERROR (2      ),ICC   (20     ),MODE  (3      )         ZDAT0010
      DIMENSION MODEB (2      ),NSAVE (20     ),PERB  (2      )         ZDAT0011
      DIMENSION STEP  (3      ),STEPB (2      ),TMINB (2      )         ZDAT0012
      DIMENSION TMIN1 (3      ),TOL   (5  ,2  )                         ZDAT0013
      EQUIVALENCE (A     ,CMA(813)),(ANGLES,CMA(786)),(ASTART,CMA(798)) ZDAT0014
      EQUIVALENCE (B     ,CMA(814)),(CLEAR ,CMA(930)),(CMAX  ,CMA(928)) ZDAT0015
      EQUIVALENCE (CONM  ,CMA(717)),(CONN  ,CMA(718)),(DELMX ,CMB(163)) ZDAT0016
      EQUIVALENCE (DELMXB,CMB(174)),(DELTBT,CMA(803)),(DELTKT,CMB(058)) ZDAT0017
      EQUIVALENCE (DELTST,CMA(931)),(DTIME ,CMA(509)),(EREF  ,CMA(713)) ZDAT0018
      EQUIVALENCE (ERLIMT,CMA(706)),(ERR   ,CMB(041)),(ERRMXK,CMB(054)) ZDAT0019
      EQUIVALENCE (ERROR ,CMB(042)),(ESTART,CMA(795)),(FM    ,CMA(715)) ZDAT0020
      EQUIVALENCE (G     ,CMA(716)),(ICC   ,CMB(201)),(IKICK ,CME(200)) ZDAT0021
      EQUIVALENCE (IMODE ,CMB(061)),(JKICK ,CMB(063)),(LAST  ,CMA(711)) ZDAT0022
      EQUIVALENCE (LAST1 ,CMA(753)),(MODE  ,CMB(160)),(MODEB ,CMB(172)) ZDAT0023
      EQUIVALENCE (MODEC ,CMB(180)),(NDAMP ,CMA(926)),(NDUMP ,CMB(057)) ZDAT0024
      EQUIVALENCE (NEQ   ,CMA(709)),(NFINAL,CMA(879)),(NPRINT,CMB(181)) ZDAT0025
      EQUIVALENCE (NSAVE ,CMB(241)),(NSHOT ,CMA(929)),(NST   ,CMA(708)) ZDAT0026
      EQUIVALENCE (NVAR  ,CMB(073)),(OBLATD,CMA(816)),(OBLATH,CMA(817)) ZDAT0027
      EQUIVALENCE (OBLATJ,CMA(818)),(OBLATN,CMA(815)),(PERB  ,CMB(055)) ZDAT0028
      EQUIVALENCE (PERIOD,CMA(910)),(RERUN ,CMA(927)),(REVOLV,CMA(799)) ZDAT0029
      EQUIVALENCE (RO    ,CMA(904)),(ROA   ,CMB(062)),(STEP  ,CMB(166)) ZDAT0030
      EQUIVALENCE (STEPB ,CMB(176)),(STEPMX,CMA(705)),(TKTIME,CMA(804)) ZDAT0031
      EQUIVALENCE (TMINB ,CMB(178)),(TMIN1 ,CMB(169)),(TOL   ,CMB(044)) ZDAT0032
      EQUIVALENCE (TSTART,CMA(796)),(VSTART,CMA(797)),(ZLAMO ,CMA(883)) ZDAT0033
      EQUIVALENCE (ZLAM4 ,CMA(884))                                     ZDAT0034
      INTEGER CMAX,STEP,STEPMX,CLEAR,STEPB,RERUN                        ZDAT0035
      DATA                                                              ZDAT0036
     1A/20925732.0/,                                                    ZDAT0037
     1(ANGLES(I),I=1,2)/28.310293,279.461759/,                          ZDAT0038
     1ASTART/0.0/,                                                      ZDAT0039
     1B/20855568.0/,                                                    ZDAT0040
     1CLEAR/1/,                                                         ZDAT0041
     1CMAX/20/,                                                         ZDAT0042
     1CONM/3.2808398/,                                                  ZDAT0043
     1CONN/0.22481905/                                                  ZDAT0044
      DATA                                                              ZDAT0045
     1(DELMX(I),I=1,3)/3*10.0/,                                         ZDAT0046
     1(DELMXB(I),I=1,2)/10.0,10.0/,                                     ZDAT0047
     1DELTBT/2.0/,                                                      ZDAT0048
     1DELTKT/.1/,                                                       ZDAT0049
     1DELTST/10.0/,                                                     ZDAT0050
     1DTIME /1.0/,                                                      ZDAT0051
     1EREF/1.E-05/,                                                     ZDAT0052
     1ERLIMT/3.0E-05/,                                                  ZDAT0053
     1ERR/1.E-04/,                                                      ZDAT0054
     1ERRMXK/1.0E-06/,                                                  ZDAT0055
     1(ERROR(I),I=1,2)/1.E-04,1.E-03/,                                  ZDAT0056
```

```
1ESTART/90.0/
 DATA
1FM/1.4076539E+16/,
1G/32.174/,
1ICC(1)/1/,
1IKICK/0/,
1IMODE/2/,
1JKICK/1/,
1LAST/5/,
1LAST1/1/,
1(MODE(I),I=1,3)/61,51,61/,
1(MODEB(I),I=1,2)/61,61/,
1MODEC/2/,
1NDAMP/5/,
1NDUMP/1/,
1NFINAL/1/,
1NEQ/9/,
1NPRINT/0/,
1NSAVE(1)/1/,
1NSHOT/0/,
1NST/1/
 DATA
1NVAR/3/,
1OBLATD/7.875E-6/,
1OBLATH/-5.75E-6/,
1OBLATJ/1.62345E-3/,
1OBLATN/1.0/,
1(PERB(J),J=1,2)/.5,.8/,
1PERIOD/0.0/,
1RERUN/0/,
1REVOLV/7.29211512E-05/,
1RO/20.90989E+6/,
1ROA/20969890.0/,
1(STEP(I),I=1,3)/3*1/,
1(STEPB(I),I=1,2)/1,1/,
1STEPMX/200/,
1TKTIME/15.0/,
1(TMINB(I),I=1,2)/2.0,2.0/,
1(TMIN1(I),I=1,3)/3*0.0/
 DATA
1(TOL(I,1),I=1,5)/5*1.E-04/,
1(TOL(I,2),I=1,5)/5*1.E-03/,
1TSTART/0.0/,
1VSTART/0.0/,
1ZLAM0/1.0/,
1ZLAM4/0.0/
 END
```

```
      SUBROUTINE DETERM(A,D,N,M)                                        ZDET0001
C                                                                       ZDET0002
C           DETERM COMPUTES THE VALUE OF DETERMINANTS.                  ZDET0003
C                                                                       ZDET0004
      COMMON/INVDET/F(10,10)                                            ZDET0005
      DIMENSION A(M,M)                                                  ZDET0006
      NA=N-1                                                            ZDET0007
      DO 1 I = 1,N                                                      ZDET0008
      DO 1 J=1,N                                                        ZDET0009
    1 F(I,J)=A(I,J)                                                     ZDET0010
      DO 5 L = 1,NA                                                     ZDET0011
      LA=L+1                                                            ZDET0012
      IF(F(L,L).NE.0.0) GO TO 4                                         ZDET0013
      DO 3  I = LA,N                                                    ZDET0014
      IF(F(I,L).EQ.0.0)GO TO 3                                          ZDET0015
      DO 2  J=L,N                                                       ZDET0016
      Z=F(L,J)                                                          ZDET0017
      F(L,J)=F(I,J)                                                     ZDET0018
    2 F(I,J)=-Z                                                         ZDET0019
      GO TO 4                                                           ZDET0020
    3 CONTINUE                                                          ZDET0021
      D=0.0                                                             ZDET0022
      RETURN                                                            ZDET0023
    4 DO 5 I = LA,N                                                     ZDET0024
      G=F(I,L)/F(L,L)                                                   ZDET0025
      DO 5 J = LA,N                                                     ZDET0026
    5 F(I,J)=F(I,J)-G*F(L,J)                                            ZDET0027
      D=1.0                                                             ZDET0028
      DO 6 I = 1,N                                                      ZDET0029
    6 D=D*F(I,I)                                                        ZDET0030
      RETURN                                                            ZDET0031
      END                                                               ZDET0032
```

```
      FUNCTION DOT(A,B)                                                 ZDOT0001
C                                                                       ZDOT0002
C     DOT COMPUTES THE DOT PRODUCT OF VECTORS A AND B.                  ZDOT0003
C                                                                       ZDOT0004
      DIMENSION A(3),B(3)                                               ZDOT0005
      DO 1 J = 1,3                                                      ZDOT0006
    1 DOT=A(1)*B(1)+A(2)*B(2)+A(3)*B(3)                                 ZDOT0007
      RETURN                                                            ZDOT0008
      END                                                               ZDOT0009
```

```
      SUBROUTINE EQUAT1
C
C           SUBROUTINE EQUAT1 CONTAINS THE EQUATIONS FOR THE
C     DERIVATIVES WHICH MUST BE EVALUATED IN ORDER TO INTEGRATE
C     THE EQUATIONS OF MOTION AND THE VARIATIONAL EQUATIONS FOR THE
C     UPPER PHASE PORTION OF THE DECK.  NOTE PAGES FOUR THROUGH SIX
C     IN PAYLOAD OPTIMIZATION OF MULTISTAGE LAUNCH VEHICLES
C     (NASA TN-3121).
C
      COMMON /CSTAR/ CMA(1000),CMB(1000)
      DIMENSION TS    (6       )
      EQUIVALENCE (CPSI   ,CMA(888)),(DOMEGA,CMA(505)),(DPHI   ,CMA(503))
      EQUIVALENCE (DR     ,CMA(502)),(DRMASS,CMA(501)),(DU     ,CMA(504))
      EQUIVALENCE (DZLAM1,CMA(506)),(DZLAM2,CMA(507)),(DZLAM3,CMA(508))
      EQUIVALENCE (FDM    ,CMA(886)),(FLOW   ,CMA(877)),(FM     ,CMA(715))
      EQUIVALENCE (G      ,CMA(716)),(IMODE  ,CMB(061)),(NSTAGE,CMA(710))
      EQUIVALENCE (TIME   ,CMA(409)),(TS     ,CMA(932)),(U      ,CMA(404))
      EQUIVALENCE (V      ,CMA(889)),(VELEX  ,CMA(870)),(ZLAM1  ,CMA(406))
      EQUIVALENCE (ZLAM2  ,CMA(407)),(ZLAM3  ,CMA(408))
C
C     CALCULATE THE SINE AND COSINE OF THE THRUST ANGLE
      IF(ZLAM2.NE.0.0) GO TO 1
      CPSI=0.0
      SPSI=SIGN(1.0,ZLAM1)
      GO TO 2
    1 TPSI=ZLAM1/ZLAM2
      TPSISQ = TPSI**2
      DENOM=SQRT(1.0+TPSISQ)
      SPSI=SIGN(TPSI/DENOM,ZLAM1)
      CPSI = SIGN(1.0/DENOM,ZLAM2)
C
    2 PSI=ARCTAN(SPSI,CPSI)
      RSQ=R**2
    3 FDM=VELEX*FLOW/RMASS
C
C     DERIVATIVE OF WEIGHT
    4 DRMASS = -FLOW
C
C     DERIVATIVES OF THE EQUATIONS OF MOTION.
      DR=U
      DPHI = OMEGA
      DU=OMEGA**2*R-FM/RSQ+FDM*SPSI
      DOMEGA = -2.0*U*OMEGA/R+FDM*CPSI/R
C
C     DERIVATIVES OF VARIATIONAL EQUATIONS.
      DZLAM1=2.0*OMEGA*ZLAM2-ZLAM3
      DZLAM2=-2.0*OMEGA*ZLAM1+U*ZLAM2/R-ZLAM4/R
      DZLAM3=DOMEGA*ZLAM2-(2.0*FM/RSQ+OMEGA**2*R)*ZLAM1/R
C
C     COMPUTE TOTAL VELOCITY
      V=SQRT(U**2+RSQ*OMEGA**2)
      RETURN
      END
```

```
      SUBROUTINE EQUAT2
C
C          SUBROUTINE EQUAT2 CONTAINS THE EQUATIONS FOR THE DERIVATIVES
C     OF THE EQUATIONS OF MOTION FOR THE BOOSTER PHASE AND ALSO CALLS
C     THE SUBROUTINES WHICH COMPUTE AERODYNAMIC AND PROPULSION
C     CHARACTERISTICS OF THE BOOSTFR SFTMENTS.
C
      COMMON /CSTAR/ CMA(1000),CMB(1000)
      COMMON /ATABLE/ CME(8000)
      DIMENSION COMPA (3       ),RB     (5       ),VATM   (5       )
      DIMENSION VX    (5       ),X      (100     ),XDOT   (100     )
      EQUIVALENCE (COMPA ,CMA(783)),(FLOW  ,CMA(752)),(GM    ,CMA(715))
      EQUIVALENCE (OBLATN,CME(002)),(Q     ,CMA(807)),(RB    ,CMA(754))
      EQUIVALENCE (VATM  ,CMA(764)),(VX    ,CMA(759)),(X     ,CMA(401))
      EQUIVALENCE (XDOT  ,CMA(501))
C
C     COMPUTE TOTAL VELOCITY AND RADIUS
      DO 1 K=1,3
      VX(K)=X(K+2)
    1 RB(K)=X(K+5)
      VX(4)=DOT(VX,VX)
      VX(5)=SQRT(VX(4))
      RB(4)=DOT(RB,RB)
      RB(5)=SQRT(RB(4))
C
C     TEST FOR OBLATENESS PERTURBATION COMPUTATION.
      IF(OBLATN.NE.0.0) CALL OBLATE
C
C
C     CALL AERODYNAMIC ROUTINES
      CALL ATMOS
      CALL AERO
C
C     CALL SIMPRO FOR THRUST AND FLOW DATA
      CALL SIMPRO
C
C     DERIVATIVE OF WEIGHT
      XDOT(2)=-FLOW
C
C     DERIVATIVES OF EQUATIONS OF MOTION IN RECTANGULAR COORDINATES.
      DO 2 K=1,3
      XDOT(K+5)=VX(K)
    2 XDOT(K+2)=COMPA(K)-GM*RB(K)/RB(5)/RB(4)
C
C     DERIVATIVE OF TOTAL HEATING PARAMETER.
      XDOT(1)=Q*VATM(5)
C
      RETURN
      END
```

```
      SUBROUTINE ERRURZ
C
C          SUBROUTINE ERRORZ IS PART OF THE INTEGRATION ERROR AND
C     STEPSIZE CONTROL PACKAGE.  SINCE A TRAJECTORY USES BOTH
C     RECTANGULAR AND POLAR COORDINATES, PROVISION IS MADE FOR
C     COMPARING THE DIFFERENCES BETWEEN THE FOURTH ORDER RUNGE-KUTTA
C     AND SIMPSON SCHEMES WITH SOME MEASURE OF THE IMPORTANCE OF THE
C     DIFFERENCE FOR BOTH OF THESE SYSTEMS.  FOR A DISCUSSION OF THE
C     INTEGRATION METHOD AND ERROR AND STEPSIZE CONTROL, SEE THE
C     N-BODY CODE - A GENERAL FORTRAN CODE FOR SOLUTION OF PROBLEMS
C     IN SPACE MECHANCIS BY NUMERICAL METHODS, BY WILLIAM C. STRACK,
C     WILBUR F. DOBSON, AND VEARL N. HUFF, NASA TN D-1455.
C
      COMMON /CSTAR/ CMA(1000),CMB(1000)
      COMMON /RUNG/RUN(125)
      DIMENSION CHECK (10      ),RELERR(100     ),XINC   (100     )
      EQUIVALENCE (A1     ,RUN(101)),(A2     ,RUN(102)),(DELT   ,CMA(701))
      EQUIVALENCE (E2     ,RUN(105)),(IMODE  ,CMB(061)),(NEQ    ,CMA(709))
      EQUIVALENCE (R      ,CMA(402)),(RB5    ,CMA(758)),(RELERR,RUN(001))
      EQUIVALENCE (RMASS  ,CMA(401)),(STEPGO,RUN(112)),(V      ,CMA(889))
      EQUIVALENCE (VX5    ,CMA(763)),(WEIGHT,CMA(402)),(XINC   ,RUN(001))
      EQUIVALENCE (ZLAM1  ,CMA(406)),(ZLAM2  ,CMA(407)),(ZLAM3  ,CMA(408))
      INTEGER STEPGO
      DATA CHECK(9)/1.0E+10/
      E2=0.0
      IF(IMODE.EQ.2) GO TO 1
C
C     COMPARISON ARRAY FOR BOOSTER INTEGRATED VARIABLES
      CHECK(2)=WEIGHT
      VX5A = VX5 + 100.0
      CHECK(3)=VX5A
      CHECK(4)=VX5A
      CHECK(5)=VX5A
      CHECK(6)=RB5
      CHECK(7)=RB5
      CHECK(8)=RB5
      NST=2
C
      GO TO 3
C
C     COMPARISON ARRAY FOR UPPER PHASE INTEGRATED VARIABLES
    1 IF(STEPGO.EQ.0) HOLD = 1.0
      IF(ABS(ZLAM3).LT.ABS(HOLD)) GO TO 2
      HOLD=ZLAM3
    2 DENOM=SQRT(ZLAM1*ZLAM1+ZLAM2*ZLAM2)
      CHECK(1)=RMASS
      CHECK(2)=R
      CHECK(3)=6.2831853
      CHECK(4)=V
      CHECK(5)=V*R
      CHECK(6)=DENOM
      CHECK(7)=DENOM
      CHECK(8)=HOLD
      CHECK(10)=DENOM
      NST=1
    3 DO 5 J = NST,NEQ
```

```
C                                                                       ZERR0057
C     DIVIDE DIFFERENCES IN TWO INTEGRATION SCHEMES BY COMPARISON       ZERR0058
C     ARRAY AND FIND LARGEST ABSOLUTE ERROR.                            ZERR0059
      RELERR(J)=XINC(J)/CHECK(J)                                        ZERR0060
C                                                                       ZERR0061
      IF(ABS(RELERR(J))-E2)      5,5,4                                  ZERR0062
    4 E2=ABS(RELERR(J))                                                 ZERR0063
    5 CONTINUE                                                          ZERR0064
C     COMPUTE FACTOR IN STEP-SIZE CONTROL EQUATION.                     ZERR0065
      E2=E2+2.E-08                                                      ZERR0066
      A1=A2                                                             ZERR0067
      A2=ALOG(E2)-5.0*ALOG(ABS(DELT))                                   ZERR0068
      RETURN                                                            ZERR0069
      END                                                               ZERR0070
```

```
      SUBROUTINE FINAL(N)
C
C          SUBROUTINE FINAL COMPUTES THE DIFFERENCES BETWEEN THE
C     ACTUAL FINAL CONDITIONS ACHIEVED IN THE INTEGRATED TRAJECTORY
C     AND THE DESIRED FINAL CONDITIONS.
C
      COMMON /CSTAR/ CMA(1000),CMB(1000)
      COMMON /FINCMP/ FNCP(6,5)
      DIMENSION CONST (5   ,2   ),FLOMX (6        ),FY     (6   ,6   )
      DIMENSION FYD    (5        ),IDATA (6   ,5   ),JFINAL(6        )
      DIMENSION PROP   (6        ),S      (6   ,2   )
      EQUIVALENCE (BETA  ,CMA(880)),(CONST ,CMA(894)),(DELTAV,CMA(861))
      EQUIVALENCE (ENERGY,CMA(892)),(FLOMX ,CMA(837)),(FM    ,CMA(715))
      EQUIVALENCE (FY    ,CMA(943)),(FYD   ,CMB(036)),(IDATA ,CMB(086))
      EQUIVALENCE (JFINAL,CMB(136)),(LAST  ,CMA(890)),(NCUTE ,CMA(893))
      EQUIVALENCE (NFINAL,CMA(879)),(NOPTA ,CMB(070)),(OMEGA ,CMA(405))
      EQUIVALENCE (PROP  ,CMA(849)),(R     ,CMA(402)),(RMASS ,CMA(401))
      EQUIVALENCE (S     ,CMB(074)),(U     ,CMA(404)),(V     ,CMA(889))
      EQUIVALENCE (VELEX ,CMA(870)),(ZLAM1 ,CMA(406)),(ZLAM2 ,CMA(407))
      EQUIVALENCE (ZLAM3 ,CMA(408))
      I = 0
      IF(NCUTE.EQ.0) I = 1
      GO TO (1,2,3,4),NFINAL
C
C     FINAL CONDITIONS FOR SPECIFIED RADIUS, RADIAL VELOCITY, AND
C     ANGULAR VELOCITY.
    1 FY(N,1)=R-FYD(1)
      FY(N,2)=U-FYD(2)
      FY(N,3)=OMEGA-FYD(3)
C
      GO TO 5
C
C     FINAL CONDITIONS FOR SPECIFIED ENERGY, WITH OPTIMIZED RADIUS AND
C     VELOCITY.
    2 FY(N,1)=ZLAM3*U-ZLAM1*(R*OMEGA**2+FM/R**2)
      FY(N,2)=ZLAM2*U-ZLAM1*OMEGA*R
      FY(N,3)=(U**2+(R*OMEGA)**2)/2.0-FM/R-ENERGY
C
      GO TO 5
C
C     FINAL CONDITIONS FOR SPECIFIED ENERGY AND FLIGHT PATH ANGLE WITH
C     OPTIMIZED RADIUS.
    3 FY(N,1)=U/R/OMEGA-SIN(BETA)/COS(BETA)
      VEL=U**2+(R*OMEGA)**2
      FY(N,2)=ZLAM3-ZLAM2*OMEGA/VEL*(FM/R+VEL)-ZLAM1*FM*U/VEL/R**2
      FY(N,3)=VEL/2.0-FM/R-ENERGY
      GO TO 5
C
C     FINAL CONDITONS FOR SPECIFIED ENERGY AND PERIGEE RADIUS WITH
C     OPTIMIZED INJECTION TRUE ANOMALY.
    4 RSQ=R**2
      P=(RSQ*OMEGA)*(RSQ*OMEGA)/FM
      E=SQRT(1.0-P*(2.0/R-V**2/FM))
      FY(N,1)=P/(1.0+E)-FYD(1)
      FY(N,2)=ZLAM1*(R*OMEGA**2-FM/RSQ)+U*(ZLAM3-2.0*OMEGA*ZLAM2)
      FY(N,3)=0.5*V**2-FM/R-ENERGY
```

```
C
C     THIS SECTION CONTAINS THE FINAL CONDITIONS FOR OPTIMIZING STAGE
C     SIZE.  THE EQUATION NUMBERS FROM PAYLOAD OPTIMIZATION OF
C     MULTISTAGE LAUNCH VEHICLES ARE LISTED WITH THE EQUATION.
    5 DO 11 J = 1,LAST
      IF(JFINAL(J).EQ.0) GO TO 11
      K = JFINAL(J)
      I = I +1
      GO TO (6,9),K
    6 IF(IDATA(J+1,5).EQ.0) GO TO 7
C
C     EQUATION 41C
      FY(N,I+2)=CONST(J,1)
C
      GO TO 11
    7 IF(J.GT.NOPTA) GO TO 8
C
C     EQUATION 48B
      FNCP(J,1)=S(J+1,1)
      FNCP(J,2)=S(J,2)/(1.0+PROP(J))
      FY(N,I+2)=FNCP(J,1)-FNCP(J,2)
      GO TO 11
C
C     EQUATION 48A
    8 FNCP(J,1)=SADDB(1,NOPTA+1,J)
      FNCP(J,2)=SADDB(2,NOPTA+1,J-1)
      FNCP(J,3)=S(NOPTA,2)/(1.0+PROP(NOPTA))
      FNCP(J,4)=S(J+1,1)
      FNCP(J,5)=S(J,2)
      FY(N,I+2)=FNCP(J,1)-FNCP(J,2)-FNCP(J,3)+FNCP(J,4)-FNCP(J,5)
      GO TO 11
C
    9 IF (J.EQ.LAST) GO TO 10
C     EQUATION 47
      FNCP(J,1)=(1.0+PROP(J))/PROP(J)
      FNCP(J,2)=S(J,1)
      FNCP(J,3)=S(NOPTA,2)/(1.0+PROP(NOPTA))
      FNCP(J,4)=SADDA(NOPTA+1,J-1)
      FNCP(J,5)=S(J,2)
      FY(N,I+2)=FNCP(J,1)*(FNCP(J,2)-FNCP(J,3)-FNCP(J,4))-FNCP(J,5)
      GO TO 11
C     EQUATION 50
   10 FNCP(J,1)=S(LAST,2)
      FNCP(J,2)=(1.0+PROP(LAST))*EXP(-DELTAV/VELEX) /PROP(LAST)
      FNCP(J,3)=S(NOPTA,2)/(1.0+PROP(NOPTA))
      FNCP(J,4)=S(LAST,1)
      FNCP(J,5)=SADDA(NOPTA+1,LAST-1)
      FY(N,I+2)=FNCP(J,1)+FNCP(J,2)*(FNCP(J,3)-FNCP(J,4)+FNCP(J,5))
   11 CONTINUE
C
C     THE FINAL WEIGHT IS STORED IN FY(N,6)
      FY(N,6)=RMASS
C
      RETURN
      END
```

```
      SUBROUTINE INVERT (A,B,N,SIZE)                                    ZINV0001
C     THIS SUBROUTINE INVERTS A NONSINGULAR NXN MATRIX.                 ZINV0002
C                                                                       ZINV0003
      INTEGER SIZE                                                      ZINV0004
      DIMENSION A(SIZE,SIZE),B(SIZE,SIZE)                               ZINV0005
C                                                                       ZINV0006
      COMMON /INVDET/F(10,10)                                           ZINV0007
C                                                                       ZINV0008
C                                                                       ZINV0009
      CALL DETERM(A,E,N,SIZE)                                           ZINV0010
      IF (E.NE.0.0) GO TO 2                                             ZINV0011
      WRITE (6,1)                                                       ZINV0012
    1 FORMAT (18HODETERMINANT = 0.0)                                    ZINV0013
      RETURN                                                            ZINV0014
C                                                                       ZINV0015
    2 DO 4 I = 1,N                                                      ZINV0016
      DO 4 J = 1,N                                                      ZINV0017
      IF(I.EQ.J)                   GO TO 3                              ZINV0018
      B(I,J)=0.0                                                        ZINV0019
      GO TO 4                                                           ZINV0020
    3 B(I,J)=1.0                                                        ZINV0021
    4 F(I,J)=A(I,J)                                                     ZINV0022
      DO 8 K=2,N                                                        ZINV0023
      L=K-1                                                             ZINV0024
      IM=L                                                              ZINV0025
      DO 5 I = L,N                                                      ZINV0026
    5 IF(ABS(F(I,L)).GT.ABS(F(IM,L)))IM=I                               ZINV0027
      IF(IM.EQ.L)     GO TO 7                                           ZINV0028
      ICOUNT=ICOUNT+1                                                   ZINV0029
      DO 6 J = 1,N                                                      ZINV0030
      G=F(IM,J)                                                         ZINV0031
      H=B(IM,J)                                                         ZINV0032
      F(IM,J)=F(L,J)                                                    ZINV0033
      B(IM,J)=B(L,J)                                                    ZINV0034
      F(L,J)=G                                                          ZINV0035
    6 B(L,J)=H                                                          ZINV0036
    7 DO 8 I=K,N                                                        ZINV0037
      C=F(I,L)/F(L,L)                                                   ZINV0038
      DO 8 J = 1,N                                                      ZINV0039
      B(I,J)=B(I,J)-C*B(L,J)                                            ZINV0040
    8 F(I,J)=F(I,J)-C*F(L,J)                                            ZINV0041
      IM=N-1                                                            ZINV0042
      DO 9 K=1,IM                                                       ZINV0043
      L=N-K+1                                                           ZINV0044
      M=N-K                                                             ZINV0045
      DO 9 I=1,M                                                        ZINV0046
      C=F(I,L)/F(L,L)                                                   ZINV0047
      DO 9 J=1,N                                                        ZINV0048
    9 B(I,J)=(B(I,J)-C*B(L,J))                                          ZINV0049
      DO 10 I = 1,N                                                     ZINV0050
      DO 10 J = 1,N                                                     ZINV0051
   10 B(I,J)=B(I,J)/F(I,I)                                              ZINV0052
      RETURN                                                            ZINV0053
      END                                                               ZINV0054
```

```
      SUBROUTINE ITERAT(X,XINC,Y,DESIRE,TOL,I,IMAX,ICOM,IFUNC,TEMP,ERR) ZITR0001
C                                                                       ZITR0002
C          SUBROUTINE ITERAT IS A GENERAL ITERATING ROUTINE.            ZITR0003
C                                                                       ZITR0004
C     THE FOLLOWING PARAMETERS ARE NECESSARY TO OPERATE THE             ZITR0005
C     ROUTINE.                                                          ZITR0006
C     X            INDEPENDENT VARIABLE                                 ZITR0007
C                                                                       ZITR0008
C     XINC         DELTAX = X*XINC FOR SECOND PASS                      ZITR0009
C                                                                       ZITR0010
C     Y            DEPENDENT VARIABLE                                   ZITR0011
C                                                                       ZITR0012
C     DESIRE       DESIRED VALUE OF Y                                   ZITR0013
      EQUIVALENCE (OMEGA ,CMA(405)),(PHI   ,CMA(403)),(PSI   ,CMA(885)) ZEQA0017
      EQUIVALENCE (R     ,CMA(402)),(RMASS ,CMA(401)),(SPSI  ,CMA(887)) ZEQA0018
C                                                                       ZITR0014
C     TOL          TOLERANCE IN Y FOR CONVERGENCE.  IF ZERO,DATA IS     ZITR0015
C                  STORED BUT NO ITERATION OCCURS.                      ZITR0016
C                                                                       ZITR0017
C     I            COUNTER INCREMENTED FOR EACH PASS THROUGH ROUTINE, I ZITR0018
C                  MUST BE ZEROED EXTERNALLY BEFORE FIRST PASS.         ZITR0019
C                                                                       ZITR0020
C     IMAX         TOTAL PASSES ALLOWED                                 ZITR0021
C                                                                       ZITR0022
C     ICOM         FLAG FOR EXTERNAL USE TO INDICATE CONVERGENCE OR IMAX ZITR0023
C                  EXCEEDED. (SEE BELOW)                                ZITR0024
C                                                                       ZITR0025
C     IFUNC        FLAG TO INDICATE CHOICE OF CONVERGENCE TECHNIQUES.   ZITR0026
C                  (SEE BELOW)                                          ZITR0027
C                                                                       ZITR0028
C     TEMP         STORAGE PROVIDED BY CALLING PROGRAM.                 ZITR0029
C                                                                       ZITR0030
C     ERR          VALUE AGAINST WHICH DEVIATION OF Y FROM DESIRE IS    ZITR0031
C                  COMPARED. (COMPARISON FUNCTION)                      ZITR0032
C                                                                       ZITR0033
C     ICOM = 1  ITERATING.                                              ZITR0034
C          = 2  CONVERGED.                                              ZITR0035
C          = 3  IMAX EXCEEDED.                                          ZITR0036
C                                                                       ZITR0037
C                                                                       ZITR0038
C                        DAMPING MODES.                                 ZITR0039
C     IFUNC = 1  LINEAR.                                                ZITR0040
C           = 2  Y=AX**2+BX+C                                           ZITR0041
C           = 3  X=AY**2+BY+C                                           ZITR0042
C                                                                       ZITR0043
C                        ELIMINATES WORST VALUE OF X.                   ZITR0044
C           = 4  LINEAR.                                                ZITR0045
C           = 5  Y=AX**2+BX+C                                           ZITR0046
C           = 6  X=AY**2+BY+C                                           ZITR0047
C                                                                       ZITR0048
      DIMENSION TEMP(5,2)                                               ZITR0049
C                                                                       ZITR0050
C     INCREMENT I                                                       ZITR0051
      I=I+1                                                             ZITR0052
C                                                                       ZITR0053
C     COMPARE DIFFERENCE WITH COMPARISON FUNCTION                       ZITR0054
```

```
      D=Y-DESIRE
      ERROR=ABS(D/ERR)
C
C     PREVIOUS ERROR STORED IN TEMP(5,1)
      CHECK=ERROR-TEMP(5,1)
C
      IF(TOL.EQ.0.0) GO TO 12
      IF(ERROR.LT.TOL) GO TO 24
      IF(I.LT.IMAX) IF(I-2) 1,2,3
      GO TO 23
C
C     COMPUTE INCREMENT IN X FOR SECOND PASS
    1 TEMP(5,2)=X*XINC
      ICOM=1
C
C
      GO TO 12
C
C     LINEAR INTERPOLATION
    2 TEMP(5,2)=D*(X-TEMP(1,1))/(TEMP(2,1)-D)
C
      GO TO 12
    3 GO TO(4,7,7,2,8,8,1),IFUNC
    4 IF(CHECK.GE.0.0) GO TO 6
    5 GO TO(2,8,10),IFUNC
    6 IF(TEMP(1,1).EQ.X) GO TO 23
C
C     EXECUTION COMES HERE IF ERROR INCREASES, DAMPS
      TEMP(5,2)=(TEMP(1,1)-X)/2.0
C
      GO TO 12
    7 IF(I.GT.3) GO TO 4
    8 D13=TEMP(2,1)-D
      D23=TEMP(2,2)-D
      F1=TEMP(1,1)-X
      F2=TEMP(1,2)-X
      GO TO (10,10,11,10,10,11),IFUNC
C
C     QUADRATIC INTERPOLATION, Y = AX**2+BX+C
   10 F12=TEMP(1,1)-TEMP(1,2)
      BOVERA=(F1*F1*D23-F2*F2*D13)/(D13*F2-D23*F1)
      RADCAL=BOVERA**2/4.0-F1*F2*F12/(D13*F2-D23*F1)*D
      IF(RADCAL.LT.0.0) GO TO 26
      A=(D13*F2-D23*F1)/(F1**2*F2-F1*F2**2)
      TEMP(5,2)=-BOVERA/2.0+SIGN(SQRT(RADCAL),(D-TEMP(2,1))*A/(X-TEMP(1,
     11)))
C
      GO TO 12
C
C     QUADRATIC INTERPOLATION, X = AY**2+BY+C
   11 D12=TEMP(2,1)-TEMP(2,2)
      TEMP(5,2)=(-TEMP(2,1)**2*F2*D+D**2*(TEMP(2,1)*F2-F1*TEMP(2,2))+
     1TEMP(2,2)**2*F1*D)/(TEMP(2,1)**2*D23-TEMP(2,2)**2*D13+D**2*D12)
C
C
C     TEMP(1,1) HAS THE BEST VALUE OF X
C     TEMP(1,2) HAS THE BEST VALUE OF Y
C     TEMP(2,1) HAS THE NEXT BEST VALUE OF X
C     TEMP(2,2) HAS THE NEXT BEST VALUE OF Y
C
```

```
   12 IF(I-2) 20,13,15                                                  ZITR0115
   13 IF(CHECK.LT.0.0) GO TO 19                                         ZITR0116
   14 TEMP(1,2)=X                                                       ZITR0117
      TEMP(2,2)=D                                                       ZITR0118
      GO TO 22                                                          ZITR0119
   15 IF(IFUNC.LT.4) IF(CHECK) 19,22,22                                 ZITR0120
   17 IF(CHECK.GE.0.0) IF(ABS(D)-ABS(TEMP(2,2))) 14,22,22               ZITR0121
   19 TEMP(1,2)=TEMP(1,1)                                               ZITR0122
      TEMP(2,2)=TEMP(2,1)                                               ZITR0123
   20 TEMP(1,1)=X                                                       ZITR0124
      TEMP(2,1)=D                                                       ZITR0125
C                                                                       ZITR0126
C     TEMP(5,1) STORES THE LEAST VALUE OF ERROR.                        ZITR0127
   21 TEMP(5,1)=ERROR                                                   ZITR0128
C                                                                       ZITR0129
   22 IF(TOL.EQ.0.0) RETURN                                             ZITR0130
C                                                                       ZITR0131
C     TEMP(5,2) CONTAINS THE CHANGE IN X.                               ZITR0132
      X=X+TEMP(5,2)                                                     ZITR0133
C                                                                       ZITR0134
      RETURN                                                            ZITR0135
   23 ICOM=3                                                            ZITR0136
      GO TO 25                                                          ZITR0137
   24 ICOM=2                                                            ZITR0138
   25 I=0                                                               ZITR0139
      RETURN                                                            ZITR0140
   26 WRITE (6,27)                                                      ZITR0141
      GO TO 23                                                          ZITR0142
   27 FORMAT (31HONEGATIVE SQUARE ROOT IN ITERAT)                       ZITR0143
      END                                                               ZITR0144
```

```
      SUBROUTINE LOAD (VAR,NVAR,NKICK)                                  ZLOA0001
C                                                                       ZLOA0002
C           SUBROUTINE LOAD TRANSFERS THE ALTITUDE, RADIAL VELOCITY,    ZLOA0003
C     AND HORIZONTAL VELOCITY INTO THE VAR ARRAY.                       ZLOA0004
      COMMON /CSTAR/ CMA(1000),CMB(1000)                                ZLOA0005
      COMMON/ATABLE/CME(8000)                                           ZLOA0006
      DIMENSION VAR(NVAR,25,NKICK)                                      ZLOA0007
      DIMENSION V      (6       )                                       ZLOA0008
      EQUIVALENCE (IBURN ,CMB(072)),(IKICK ,CME(200)),(V      ,CMB(154)) ZLOA0009
      IF(IBURN.EQ.IBURNT.OR.IBURN.EQ.0) RETURN                          ZLOA0010
      IBURNT=IBURN                                                      ZLOA0011
      DO 1 J = 1,NVAR                                                   ZLOA0012
    1 VAR(J,IBURN,IKICK)=V(J)                                           ZLOA0013
      RETURN                                                            ZLOA0014
      END                                                               ZLOA0015
```

```
C     MAIN
C
C          MAIN IS THE CONTROLLING ROUTINE FOR THE DYNAMIC TABLE
C     PROGRAM.  IT IS BASICALLY A GENERAL NXN NEWTON-RAPHSON SCHEME.
C
      COMMON /CSTAR/ CMA(1000),CMB(1000)
      COMMON /ATABLE/ CMF(8000)
      COMMON /PERB/NA,N,SF(5),COUNT,RSTO
      DIMENSION COMP   (5       ),CONST (5   ,2   ),DELFY (6   ,5   )
      DIMENSION DELFYD(5        ),DELTEM(5   ,5   ),DELX  (5   ,5   )
      DIMENSION DELXD (5        ),DELXIN(5   ,5   ),DELXO (6   ,5   )
      DIMENSION DFYIN (5   ,5   ),DX     (5       ),DYDXIN(5   ,5   )
      DIMENSION ERROR (2        ),EVSQ   (6       ),FLOMX (6        )
      DIMENSION FORCES(6        ),FY     (6   ,6  ),NOPT  (6        )
      DIMENSION NOPTST(6        ),SDEL   (5   ,5  ),SDELN (5   ,5   )
      DIMENSION TABLE (300      ),THRUST(6        ),TOL   (5   ,2   )
      DIMENSION VELEXP(6        ),WTFLOW(6        ),XINPT (100      )
      DIMENSION XO     (6   ,5  ),XPRIM (100,2   )
      EQUIVALENCE (CLEAR ,CMA(930)),(CMAX  ,CMA(928)),(COMP  ,CMB(031))
      EQUIVALENCE (CONST ,CMA(894)),(DELTK ,CMF(199)),(DELTKT,CMB(058))
      EQUIVALENCE (DELTST,CMA(931)),(DX    ,CMA(938)),(EREF  ,CMA(713))
      EQUIVALENCE (ERLOG ,CMA(707)),(ERR   ,CMB(041)),(ERRMXK,CMB(054))
      EQUIVALENCE (ERROR ,CMB(042)),(FIXDTK,CMB(071)),(FLOMX ,CMA(837))
      EQUIVALENCE (FORCES,CMA(739)),(FY    ,CMA(943)),(G     ,CMA(716))
      EQUIVALENCE (IMODE ,CMB(061)),(ITERP ,CMB(069)),(JDATA ,CMA(925))
      EQUIVALENCE (JKICK ,CMB(063)),(LAST  ,CMA(890)),(LAST1 ,CMA(753))
      EQUIVALENCE (MASH  ,CMB(064)),(NDAMP ,CMA(926)),(NDUMP ,CMB(057))
      EQUIVALENCE (NEQ   ,CMA(709)),(NOPT  ,CMA(819)),(NSHOT ,CMA(929))
      EQUIVALENCE (NST   ,CMA(708)),(NSTAGE,CMA(710)),(R     ,CMA(402))
      EQUIVALENCE (RERUN ,CMA(927)),(THRUST,CMA(831)),(TOL   ,CMB(044))
      EQUIVALENCE (VELEX ,CMA(870)),(VELEXP,CMA(864)),(WTFLOW,CMA(733))
      EQUIVALENCE (XINPT ,CMA(601)),(XO    ,CMB(001)),(XPRIM ,CMA(001))
      DOUBLE PRECISION XPRIM
      INTEGER COUNT,CMAX,FIXDTK,RERUN,CLEAR
      EXTERNAL EQUAT1,OUTPT1
    1 WRITE (6,2)
    2 FORMAT (1H1)
      JDATA=1
      MCOUNT=0
      IF(CLEAR.EQ.1) CALL STDATA
C
C     (1)  ALL DATA EXCEPT BINARY DATA IS INTRODUCED THROUGH THE
C     INPUT ROUTINE.
      CALL INPUT (01,CMA,TABLE)
C
C     (2)  COEFNT GENERATES ANY NEEDED CURVE FITS.
      CALL COEFNT
C
C     (3)  PRINT OUT HEADINGS ARE PRINTED OUT FOR THE UPPER PHASES.
      CALL OUTPT1(0)
C
C     (4)  DELTK (GRID SPACING FOR KICK ANGLES) IS SET EQUAL TO AN
C     INPUT VALUE.
      DELTK=DELTKT
C
C
```

```
C      (5)  TAKE LOG OF EREF FOR USE IN STEP.
       ERLOG = ALOG(EREF)
C
C      (6)  SET THRSUT AND WTFLOW OF LAST SEGMENT OF BOOSTER
       FORCES(LAST1)=THRUST(1)
       WTFLOW(LAST1)=FLOMX(1)
C
C      (7) COMPUTE JET VELOCITIES
       DO 4 J = 1,6
       IF(FLOMX(J).NE.0.0) GO TO 3
       VELEXP(J)=0.0
       GO TO 4
     3 VELEXP(J)=G*THRUST(J)/FLOMX(J)
     4 CONTINUE
C
C      (8)  IF IMODE = 3, A BINARY BOOSTER TABLE IS IN THE DATA
C      AND MUST BE READ IN.  RENDER(1) CALLS BCREAD.  OUTPT1 PRINTS
C      A GLOSSARY OF THE PROPULSION CHARACTERISTICS AND HARDWARE
C      RELATED DATA FOR EACH PHASE.
       IF(IMODE.NE.3) GO TO 5
       IMODE=2
       JKICK = 1
       CALL RENDER (1)
     5 CALL OUTPT1(2)
C
C      (9)  GUESS IS A DUMMY ROUTINE AVAILABLE FOR ANY PURPOSE
C      REQUIRED AT THE BEGINNING OF A PROBLEM.
     6 CALL GUESS
C
C      (10) XOLOAD SETS UP THE XO LIST AND DETERMINES ITERP (THE
C      VARIABLE DETERMINING THE SIZE OF THE ITERATION LOOP).
       CALL XOLOAD
C
C      (11) ASSIGN  43 TO NGO SETS NGO FOR THE STATEMENTS AT OR
C      NEAR STATEMENT 38.
       ASSIGN 43 TO NGO
C
C      (12) SEE COMMENT 35.
       COUNT=0
C
C      (13) NA AND NB ARE RELATED TO ITERATION LOOP SIZE.  SEE
C      COMMENT 10.
       NA=ITERP+2
       NB = NA + 1
C
C      (14) ND AND NREP ARE INITIALIZED.  SEE COMMENT 46.
       ND=0
       NREP=1
C
C      (15) SCOMP COMPUTES VALUES AGAINST WHICH DEVIATIONS FROM THE
C      DESIRED FINAL CONDITIONS ARE COMPARED.
       CALL SCOMP(1)
C
C      (16) IF NSHOT IS NON-ZERO, THEN THE NUMBER OF TRAJECTORIES
C      INTEGRATED WILL BE LIMITED TO THE VALUE OF NSHOT.  IF NSHOT
C      IS NON-ZERO, IT IS USUALLY EQUAL TO ONE.
       IF(NSHOT.EQ.0) GO TO 7
C
       NC=NSHOT
       GO TO 8
```

```
    7 NC=NB
    8 DO 9 J=1,NA
      DO 9 N=2,NB
C
C     (17) THE XO LIST CONTAINS THE INITIAL CONDITIONS WHICH ARE
C     VARIED TO SATISFY THE FINAL CONDITIONS.  THE XO(J+1,J),
C     J = 1,NA CONTAINS THE PERTURBATION MADE ON EACH INITIAL
C     CONDITION.
    9 XO(N,J)=XO(1,J)
      DO 10 J = 1,NA
   10 XO(J+1,J)=XO(1,J)+XO(1,J)*DX(J)
C
C     (18) ICHECK IS A COUNTER WHICH IS INCREMENTED IN PERTB AND
C     INDICATES THE NUMBER OF ATTEMPTS TO MODIFY THE PERTURBATION
C     SIZE.  SEE PERTB.
   11 ICHECK=0
C
      DO 18  N=NREP,NC
C
C     (19) START INITIALIZES THE NECESSARY VARIABLES FOR THE IN-
C     TEGRATION AND CALLS MAINA WHICH ACQUIRES THE PERTINENT BOOSTER
C     DATA.
   12 CALL START(N)
C
C     (20) MASH = 0 INDICATES THAT THE DESIRED TRAJECTORY HAS NOT
C     VIOLATED ANY OF THE INITIAL CONSTRAINTS.  IF THE FIRST
C     TRAJECTORY VIOLATES ANY CONSTRAINTS (N = 1), THEN A NEW CASE
C     IS CONSIDERED.  IF THE FIRST CASE IS ACCEPTABLE AND A PERT-
C     URBATION IS UNACCEPTABLE, THE PERTURBATION SIZE IS HALVED
C     AND THE TRAJECTORY IS ATTEMPTED AGAIN.
      IF(MASH.EQ.0) GO TO 13
      MASH=0
      IF(N.EQ.1) GO TO 1
      DX(N-1)=DX(N-1)/2.0
      GO TO 17
C
C
C     (21) XINPT CONTAINS THE INITIAL CONDITIONS.  THESE ARE
C     TRANSFERED TO THE XPRIM LIST TO BE INTEGRATED.
   13 DO 14 J = NST,NEQ
   14 XPRIM(J,1)=DBLE(XINPT(J))
C
C
C     (22) DAMODE CHOOSES THE INITIAL STEPSIZE AND INITIALIZES THE
C     OUTPUT CONTROL VARIABLES UTILIZED IN STEP.
      CALL DAMODE(1,N)
C
C     (23) RUNGEK INTEGRATES THE XINPT LIST WITH RESPECT TO TIME.
      CALL RUNGEK (EQUAT1,OUTPT1)
C
C     SEE COMMENT 20
      IF(MASH.EQ.0) GO TO 15
      MASH = 0
      IF (N.EQ.1) GO TO 57
      DX(N-1)=DX(N-1)/2.0
      ICHECK = 0
      GO TO 17
C
C     (24) FINAL COMPUTES THE DIFFERENCES (FY) BETWEEN THE DESIRED FINAL
C     CONDITIONS AND THE ONES ACHIEVED IN THE INTEGRATED TRAJECTORY.
```

```
   15 CALL FINAL(N)
C
      IF(N.GT.1) GO TO 16
      RSTO=R
      GO TO 18
C
C     (25) PERTB CHECKS THE PERTURBATION SIZE.  IF ICHECK = 0, THEN
C     THE PERTURBATION SIZE IS WITHIN THE REQUIRED BOUNDARIES.
   16 CALL PERTB(DX(N-1),ICHECK)
C
C     SEE COMMENT 20
      IF(ICHECK.EQ.0) GO TO 18
   17 XO(N,N-1)=XO(1,N-1)+XO(1,N-1)*DX(N-1)
C
      GO TO 12
   18 CONTINUE
      IF(NB.NE.NC) GO TO 51
C
C     (26) CALCULATE THE MATRIX OF DIFFERENCES IN FINAL CONDITIONS AND
C     ITS INVERSE.
   19 DO 20 N = 1,NA
      DO 20 K = 1,NA
   20 DELX(N,K)=FY(N+1,K)-FY(1,K)
      CALL INVERT(DELX,DELXIN,NA,5)
C
C     (27) CALCULATE ERROR INDICATOR FOR EACH TRAJECTORY.  SEE EQUATION
C     64 ON PAGE 26 OF PAYLOAD OPTIMIZATION OF MULTISTAGE LAUNCH
C     VEHICLES, (NASA TN-3191).
      DO 21  I=1,NA
      SF(I)=0.0
      DO 21 J = 1,NA
   21 SF(I)=DELXIN(I,J)*(FY(J+1,6)-FY(1,6))+SF(I)
      DO 22  N=1,NB
      EVSQ(N)=0.0
      DO 22  J=1,NA
   22 EVSQ(N)=EVSQ(N)+(FY(N,J)*SF(J))**2
C
C     (28) DETERMINE TRAJECTORIES WITH LARGEST AND SMALLEST ERROR
C     FMIN    ERROR FOR BEST TRAJECTORY
C     KMIN    NUMBER OF BEST TRAJECTORY
C     FMAX    ERROR FOR WORST TRAJECTORY
C     KMAX    NUMBER OF WORST TRAJECTORY
      FMIN=EVSQ(1)
      KMIN=1
      DO 23  N=2,NB
      IF(EVSQ(N).GT.EVSQ(KMIN)) GO TO 23
      FMIN=EVSQ(N)
      KMIN=N
   23 CONTINUE
      FMAX=EVSQ(1)
      KMAX=1
      DO 24  N=2,NB
      IF(EVSQ(N).LT.EVSQ(KMAX)) GO TO 24
      FMAX=EVSQ(N)
      KMAX=N
   24 CONTINUE
C
C     (29) THE KMIN TRAJECTORY IS USED AS A REFERENCE TO GENERATE
C     A SET OF PARTIAL DERIVATIVES AND INITIAL CONDITIONS ARE
C     COMPUTED FOR A NEW REFERENCE TRAJECTORY.
```

```
C     SEE SECTION ON PROCEDURE AND IMPLEMENTATION ON PAGE 25 OF PAYLOAD ZMAI0237
C     OPTIMIZATION OF MULTISTAGE LAUNCH VEHICLES, (NASA TN-3191).       ZMAI0238
      DO 25  N=1,NB                                                     ZMAI0239
      DO 25  M=1,NA                                                     ZMAI0240
      DELXO(N,M)=XO(N,M)-XO(KMIN,M)                                     ZMAI0241
   25 DELFY(N,M)=FY(N,M)-FY(KMIN,M)                                     ZMAI0242
      DO 26  I=1,NA                                                     ZMAI0243
      DO 26  J=1,NA                                                     ZMAI0244
   26 DELTEM(I,J)=DELFY(I,J)                                            ZMAI0245
      DO 27  N=KMIN,NA                                                  ZMAI0246
      DO 27  M=1,NA                                                     ZMAI0247
      DELXO(N,M)=DELXO(N+1,M)                                           ZMAI0248
   27 DELTEM(N,M)=DELFY(N+1,M)                                          ZMAI0249
      CALL INVERT (DELTEM,DFYIN,NA,5)                                   ZMAI0250
      DO 28  N=1,NA                                                     ZMAI0251
      DO 28  J=1,NA                                                     ZMAI0252
      DYDXIN(N,J)=0.0                                                   ZMAI0253
      DO 28  M=1,NA                                                     ZMAI0254
   28 DYDXIN(N,J)=DFYIN(N,M)*DELXO(M,J)+DYDXIN(N,J)                     ZMAI0255
      DO 29  N=1,NA                                                     ZMAI0256
   29 DELFYD(N)=-FY(KMIN,N)                                             ZMAI0257
      DO 30  J=1,NA                                                     ZMAI0258
      DELXD(J)=0.0                                                      ZMAI0259
      DO 30  L=1,NA                                                     ZMAI0260
   30 DELXD(J)=DELFYD(L)*DYDXIN(L,J)+DELXD(J)                           ZMAI0261
   31 DO 32  N=1,NA                                                     ZMAI0262
   32 XO(KMAX,N)=XO(KMIN,N)+DELXD(N)                                    ZMAI0263
C                                                                       ZMAI0264
C     (30) THE NEW REFERENCE TRAJECTORY IS INTEGRATED, ITS INITIAL      ZMAI0265
C     AND FINAL CONDITIONS ARE PLACED IN THE XO AND FY LIST PREVIOUSLY  ZMAI0266
C     OCCUPIED BY THE KMAX TRAJECTORY.  SEE EQUATION 63 ON PAGE 26 OF   ZMAI0267
C     PAYLOAD OPTIMIZATION OF MULTISTAGE LAUNCH VEHICLES,               ZMAI0268
C     (NASA TN-3191).                                                   ZMAI0269
      ND=0                                                              ZMAI0270
C                                                                       ZMAI0271
C     SEE COMMENT 19                                                    ZMAI0272
   33 CALL START(KMAX)                                                  ZMAI0273
C                                                                       ZMAI0274
C     SEE COMMENT 20                                                    ZMAI0275
      IF(MASH.EQ.1) GO TO 52                                            ZMAI0276
C                                                                       ZMAI0277
C     SEE COMMENT 21                                                    ZMAI0278
      DO 34  J=NST,NEQ                                                  ZMAI0279
   34 XPRIM(J,1)=DBLE(XINPT(J))                                         ZMAI0280
C                                                                       ZMAI0281
      K=0                                                               ZMAI0282
C                                                                       ZMAI0283
C     SEE COMMENT 22                                                    ZMAI0284
      CALL DAMODE (JDATA,1)                                             ZMAI0285
C                                                                       ZMAI0286
C     SEE COMMENT 23                                                    ZMAI0287
      CALL RUNGEK (EQUAT1,OUTPT1)                                       ZMAI0288
C                                                                       ZMAI0289
C                                                                       ZMAI0290
C     SEE COMMENT 20                                                    ZMAI0291
      IF (MASH.EQ.1) GO TO 52                                           ZMAI0292
C                                                                       ZMAI0293
C     (31) JDATA = 1 WHEN CONVERGENCE HAS NOT BEEN OBTAINED.  JDATA =   ZMAI0294
C     2 IMPLIES THAT THE CASE IN CONVERGED, JDATA = 3 IS USED WHEN      ZMAI0295
C     A PRESCRIBED PROPELLANT LOADING HAS BEEN EXCEEDED AND THE         ZMAI0296
```

```
C     CONVERGED CASE WITH THE EXCESS PROPELLANT IS PRINTED BEFORE       ZMAI0297
C     THE OFFENDING PROPELLANT LOADING IS FIXED AT MAXIMUM PROPELLANT   ZMAI0298
C     AND RECONVERGED.                                                  ZMAI0299
      IF(JDATA.NE.2) GO TO 35                                           ZMAI0300
C                                                                       ZMAI0301
C     (32) IF RERUN = 1, THE BOOSTER WITH THE CONVERGED KICK ANGLE AND  ZMAI0302
C     BURNING TIME IS INTEGRATED AND ITS ERROR COMPUTED.  IF THE ERROR  ZMAI0303
C     IS EXCEEDED, CONTROL PASSES TO STATEMENT  54 WHERE MCOUNT IS      ZMAI0304
C     INCREMENTED, MASH IS SET EQUAL TO ONE WHICH CAUSES DELTK TO BE    ZMAI0305
C     HALVED IN MAINA.  SEE COMMENT 49.                                 ZMAI0306
      IF(RERUN.EQ.0) GO TO 51                                           ZMAI0307
      ASSIGN 54 TO NGO                                                  ZMAI0308
      GO TO 36                                                          ZMAI0309
C                                                                       ZMAI0310
   35 IF(JDATA.EQ.1) GO TO 36                                           ZMAI0311
C     (33) CHECK IS CALLED HERE FOR THE SECOND TIME.  THE CONVERGED     ZMAI0312
C     CASE WITH THE EXCEEDED PROPELLANT LOADING HAS BEEN PRINTED.       ZMAI0313
C     CHECK COMPUTES THE FIXED PHASE TIMES FOR THE OFFENDING PRO-       ZMAI0314
C     PELLANT LOADINGS AND SETS NOPT FOR THOSE PHASES TO ZERO.          ZMAI0315
      CALL CHECK                                                        ZMAI0316
      JDATA=1                                                           ZMAI0317
      GO TO 6                                                           ZMAI0318
C                                                                       ZMAI0319
C     (34) SEE COMMENTS 24, 26, AND 27.                                 ZMAI0320
   36 CALL FINAL(KMAX)                                                  ZMAI0321
      CALL SCOMP(2)                                                     ZMAI0322
      EVP=0.0                                                           ZMAI0323
      DO 37  J=1,NA                                                     ZMAI0324
   37 EVP=EVP+(FY(KMAX,J)*SF(J))**2                                     ZMAI0325
C                                                                       ZMAI0326
C     (35) COUNT IS INCREMENTED WHEN A NEW REFERENCE TRAJECTORY IS      ZMAI0327
C     INTEGRATED.  IF COUNT EXCEEDS CMAX, THE ITERATION IS ABANDONED    ZMAI0328
C     AND NEW DATA IS CALLED IN.                                        ZMAI0329
      COUNT=COUNT+1                                                     ZMAI0330
      IF(COUNT.GT.CMAX) GO TO 51                                        ZMAI0331
C                                                                       ZMAI0332
C     (36) FIRST CONVERGENCE TEST .  SQRT (EVSQ) IS A MEASURE OF THE    ZMAI0333
C     CHANGE IN FINAL WEIGHT REQUIRED TO SATISFY THE FINAL CONDITIONS   ZMAI0334
C     BASED ON THE PARTIALS ALREADY OBTAINED.  IF THE ERROR IS LESS THANZMAI0335
C     SOME DECIMAL FRACTION OF THE INITIAL WEIGHT, THE CONTROL PASSES TOZMAI0336
C     THE NEXT CONVERGENCE TEST.  OTHERWISE ITERATION PROCEEDS UNTIL    ZMAI0337
C     COUNT EXCEEDS CMAX.  SEE EQUATION 64 ON PAGE 26 OF PAYLOAD        ZMAI0338
C     OPTIMIZATION OF MULTISTAGE LAUNCH VEHICLES, (NASA TN-3191).  IF   ZMAI0339
C     RERUN = 1,THEN AFTER CONVERGENCE THE EXACT BOOSTER IS INTEGRATED  ZMAI0340
C     AND THE UPPER PHASES ARE REINTEGRATED MAINTAINING THE SAME BURNINGZMAI0341
C     TIMES AND INITIAL THRUST ANGLES AND RATE.  THE ERROR FOR THAT     ZMAI0342
C     TRAJECTORY IS EXAMINED UTILIZING THESE EQUATIONS.                 ZMAI0343
      EVSQ(KMAX)=EVP                                                    ZMAI0344
      EVPSRT=SQRT(EVP)                                                  ZMAI0345
      IF(SQRT(EVSQ(KMAX)).GT.ERROR(JDATA)*XINPT(1)) GO TO NGO,(43,54)   ZMAI0346
C                                                                       ZMAI0347
C     (37) SECOND CONVERGENCE TEST.  IF THE ERROR IN DESIRED FINAL      ZMAI0348
C     CONDITIONS IS LESS THAN SOME DECIMAL FRACTION SPECIFIED IN TOL,   ZMAI0349
C     THE CONVERGENCE IS RECOGNIZED AND THE CONVERGED CASE IS PRINTED.  ZMAI0350
      DO 38  J=1,NA                                                     ZMAI0351
   38 IF(ABS(FY(KMAX,J)*COMP(J)).GT.TOL(J,JDATA)) GO TO NGO,(43,54)     ZMAI0352
C                                                                       ZMAI0353
C     (38) AT THIS POINT, IF JDATA ALREADY EQUALS TWO, THEN THE GRID    ZMAI0354
C     SPACING FOR THE TABLE HAS PROVED SUFFICIENTLY FINE AND A NEW      ZMAI0355
C     CASE WILL BE READ IN.  IF JDATA = 1, THEN JDATA IS SET TO TWO     ZMAI0356
```

```
C       AND THE CONVERGED CASE WILL BE PRINTED OUT.
        IF(JDATA.EQ.2) GO TO 51
        JDATA=2
C
C       (39) CHECK IS CALLED HERE FOR THE FIRST TIME IN A CASE.  IF THE
C       PROPELLANT LOADING FOR ANY PHASE EXCEEDED THE PRESCRIBED VALUE,
C       JDATA IS SET EQUAL TO THREE AND THE OFFENDING CASE IS PRINTED FOR
C       REFERENCE.
        DO 39  J = 1,LAST
     39 NOPTST(J)=NOPT(J)
        CALL CHECK
        DO 40  J=1,LAST
     40 IF(NOPT(J).NE.NOPTST(J)) GO TO 41
C
C       (40) RERUN = 1 INDICATES A CHECK ON THE TKICK GRID SPACING.
C       FIXDT = 1 IS A FLAG WHICH INDICATES TO START THAT A NON-
C       GRID KICK ANGLE AND SPECIFIC BURNING TIME WILL BE INTEGRATED.
        IF(RERUN.EQ.1) FIXDTK=1
        GO TO 33
C       (41) SEE COMMENT 39.
     41 JDATA=3
        DO 42  J = 1,LAST
     42 NOPT(J)=NOPTST(J)
        GO TO 33
C
C       (42) EVSQ(KMAX) IS THE ERROR FOR THE NEW REFERENCE(SEE COMMENT
C       30).  IF THE NEW REFERENCE IS NOT BETTER THAN THE PREVIOUS
C       ONE (KMIN), THEN IT IS DESIRABLE TO CUT THE CHANGES IN THE
C       INITIAL CONDITIONS.  SEE COMMENT 46.
     43 IF(EVSQ(KMAX).GE.EVSQ(KMIN)) GO TO 48
        EMAX=AMAX1(EVSQ(1),EVSQ(2),EVSQ(3),EVSQ(4),EVSQ(5),EVSQ(6))
        IF(SQRT(ZMIN(EVSQ,NB)).GT.ERR*XINPT(1)) GO TO 44
C
C       (43) IF THE ERROR OF THE NEW REFERENCE INDICATES THAT
C       CONVERGENCE IS CLOSE, THE NEW REFERENCE MAY REPLACE THE WORST
C       CASE AND A NEW REFERENCE COMPUTED WITHOUT REPERTURBING.
        GO TO 19
C
C       (44) IF PERTURBATIONS ON THE NEW REFERENCE ARE DESIRED,
C       THE XO LIST IS SET UP AND CONTROL IS RETURNED TO STATEMENT
C       11.
     44 DO 45  J=1,NA
        DO 45  N=1,NB
     45 XO(N,J)=XO(KMAX,J)
        DO 46  J=1,NA
     46 XO(J+1,J)=XO(1,J)+XO(1,J)*DX(J)
        NREP=2
        DO 47  K = 1,NA
     47 FY(1,K)=FY(KMAX,K)
        FY(1,6) = FY(KMAX,6)
        EVSQ(1)=EVP
        GO TO 11
C
C       (45) IF THE FIRST REFERENCE TRAJECTORY IS BETTER THAN THE SECOND,
C       THE ERROR ON THE FIRST IS CHECKED TO DETERMINE IF IT IS WITHIN
C       THE CONVERGENCE TOLERANCE.  IF IT IS NOT WITHIN TOLERANCE, THE
C       CHANGES IN INITIAL CONDITIONS FROM THE FIRST REFERENCE TO THE
C       SECOND ARE HALVED AND A NEW REFERENCE IS CONSIDERED.
     48 IF(COUNT.EQ.1.AND.SQRT(ZMIN(EVSQ,NB)).GT.ERR*XINPT(1)) GO TO 44
C       (46) SEE COMMENT 42.  IF THE ERROR FOR THE NEW REFERENCE IS
```

```
C     GREATER THAN THE ERROR FOR THE OLD REFFRENCE, THE CHANGES          ZMAI0417
C     FROM THE OLD TO THE NEW ARE HALVED UNTIL ND IS GREATER THAN        ZMAI0418
C     NDAMP OR THE NEW REFERENCE ERROR IS LESS THAN THE OLD.             ZMAI0419
      ND=ND+1                                                             ZMAI0420
      IF(ND.GT.NDAMP) GO TO 51                                            ZMAI0421
      IF(COUNT.NE.1) GO TO 49                                             ZMAI0422
      KMAX=KMIN                                                           ZMAI0423
      GO TO 44                                                            ZMAI0424
   49 DO 50  N=1,NA                                                       ZMAI0425
   50 XO(KMAX,N)=XO(KMIN,N)+DELXD(N)/2.0**ND                              ZMAI0426
      GO TO 33                                                            ZMAI0427
C                                                                         ZMAI0428
C     (47) A DUMMY ROUTINE AVAILABLE FOR USE AT THE END OF A              ZMAI0429
C     PROBLEM.                                                            ZMAI0430
   51 CALL STGSS(KMAX)                                                    ZMAI0431
C                                                                         ZMAI0432
C     RENDER PUNCHES OUT BINARY BOOSTER DATA.                             ZMAI0433
      IF(NDUMP.EQ.1) CALL RENDER (2)                                      ZMAI0434
C                                                                         ZMAI0435
      GO TO 1                                                             ZMAI0436
C                                                                         ZMAI0437
C     (48) IF MASH IS SET TO ONE IN SOME ROUTINE OTHER THAN MAIN,         ZMAI0438
C     THE CHANGE FROM NEW REFERENCE TO OLD IS HALVED AND A NEW            ZMAI0439
C     REFERENCE IS CONSIDERED.                                            ZMAI0440
   52 MASH=0                                                              ZMAI0441
      DO 53 J = 1,NA                                                      ZMAI0442
   53 DELXD(J)=DELXD(J)/2.0                                               ZMAI0443
      GO TO 31                                                            ZMAI0444
C                                                                         ZMAI0445
C     (49) IF MASH IS SET EQUAL TO ONE IN MAIN, IT IS A FLAG TO           ZMAI0446
C     MAINA TO HALVE DELTK, THE TKICK GRID SPACING.  IF AFTER THE         ZMAI0447
C     GRID SPACING IS CUT IN HALF THE NEW ERROR IS NOT LESS THAN 75       ZMAI0448
C     PERCENT OF THE OLD ERROR, THE PROGRAM TERMINATES THE PROBLEM AND    ZMAI0449
C     CALLS IN NEW DATA.                                                  ZMAI0450
   54 MASH=1                                                              ZMAI0451
      MCOUNT=MCOUNT+1                                                     ZMAI0452
      IF(MCOUNT.EQ.1.OR.EVSQ(KMAX).LT.0.75*EVSQST) GO TO 56               ZMAI0453
      WRITE (6,55)                                                        ZMAI0454
   55 FORMAT(48HONO IMPROVEMENT WITH CUT IN KICK ANGLE INCREMENT)         ZMAI0455
      MASH = 0                                                            ZMAI0456
      GO TO 51                                                            ZMAI0457
   56 EVSQST=EVSQ(KMAX)                                                   ZMAI0458
      JDATA=1                                                             ZMAI0459
      GO TO 6                                                             ZMAI0460
C                                                                         ZMAI0461
C     SEE COMMENT 20                                                      ZMAI0462
   57 WRITE (6,58) NSTAGE                                                 ZMAI0463
      GO TO 1                                                             ZMAI0464
   58 FORMAT (42HOINITIAL CONDITIONS PROHIBIT USE OF STAGE I2)            ZMAI0465
      END                                                                 ZMAI0466
```

```
      SUBROUTINE MAINA
C
C          SUBROUTINE MAINA CONTROLS THE CHOICE OF KICK ANGLES AND
C     BOOSTER BURNING TIMES UTILIZED IN THE BOOST SUBROUTINE
C     INTERPOLATION SCHEME TO ACQUIRE BOOSTER BURNOUT CONDITIONS
C     FOR A SPECIFIC KICK ANGLE AND BOOSTER BURNING TIME.
C
      COMMON /CSTAR/ CMA(1000),CMB(1000)
      COMMON/ATABLE/CME(8000)
      DIMENSION APT    (3   ,100),APTMAX(3        ),BRUN  (3        )
      DIMENSION ITB    (3        ),ITK    (3        ),KPT   (3   ,100)
      DIMENSION RK     (5        ),RKA    (5        ),TB    (6        )
      DIMENSION TBOOST(6        )
      EQUIVALENCE (APT    ,CME(201)),(APTMAX,CMB(126)),(DELMAX,CMA(702))
      EQUIVALENCE (DELTB  ,CME(016)),(DELTK  ,CME(199)),(DELTKT,CMB(058))
      EQUIVALENCE (ELEV   ,CMA(790)),(IBURN  ,CMB(072)),(IKICK  ,CME(200))
      EQUIVALENCE (IMODE  ,CMB(061)),(ITB    ,CMB(123)),(ITK    ,CMB(120))
      EQUIVALENCE (JKICK  ,CMB(063)),(KPT    ,CME(201)),(LAST   ,CMA(711))
      EQUIVALENCE (LAST1  ,CMA(753)),(MASH   ,CMB(064)),(MODEC  ,CMB(180))
      EQUIVALENCE (MODOUT,CMA(714)),(NKICKS,CME(021)),(NVAR   ,CMB(073))
      EQUIVALENCE (STEPS  ,CMA(704)),(TB     ,CMA(825)),(TBLAST,CME(018))
      EQUIVALENCE (TBOOST,CMA(745)),(TBURN  ,CMB(065)),(TKICK  ,CMB(059))
      EQUIVALENCE (TMIN   ,CMA(703)),(TMINST,CME(019)),(TSPM   ,CME(010))
      EQUIVALENCE (VAR    ,CME(501))
      LOGICAL ALOG
      INTEGER STEPS
C
C     IKICK EQUALS THE NUMBER OF DIFFERENT KICK ANGLES WHOSE
C     TRAJECTORIES HAVE BEEN RUN.  THESE ARE ARRANGED IN ASCENDING
C     ORDER IN APT(1,J).  IF NEW KICK ANGLES ARE TO BE RUN, THESE
C     ARE PLACED IN BRUN(J) IN ASCENDING ORDER.  DELTK IS THE SPACING
C     FOR THE KICK ANGLE GRID.  NRUN EQUALS THE NUMBER OF NEW
C     TRAJECTORIES REQUIRED IN ANY GIVEN PASS THROUGH THE ROUTINE.
C
C     IF THE KICK ANGLE PREDICTED IN THE COURSE OF ITERATION IS
C     GREATER THAN 90 DEGREES (WHICH IS AN UNACCEPTABLE KICK ANGLE BY
C     DEFINITION), MAINA SETS MASH  = 1, AND CONTROL EVENTUALLY BUT
C     RAPIDLY RETURNS TO MAIN WHERE THE PREDICTED KICK ANGLE IS
C     DECREASED TO UNDER 90 DEGREES.
      IF(TKICK.GE.90.0) GO TO 33
C
      IF(MASH.EQ.0) GO TO 2
C
C     MASH NOT EQUAL TO ONE IMPLIES THAT THE GRID SPACING HAS BEEN
C     FOUND TO BE TOO LARGE IN MAIN AND DELTK IS HALVED.
      MASH=0
      NRUN=1
      DELTK=DELTK/2.0
C
C     CHOOSE NEW KICK ANGLE
      IF(JKICK.EQ.1.OR.(JKICK.EQ.IKICK.AND.TKICK.GT.APT(1,JKICK)))
     1GO TO 1
      BRUN(1)=(APT(1,JKICK)+APT(1,JKICK-1))/2.0
      GO TO 17
```

```
    1 BRUN(1)=APT(1,JKICK)+SIGN(DELTK,FLOAT(JKICK)-FLOAT(IKICK)/2.0)     ZMAI0055
      GO TO 17                                                          ZMAI0056
C                                                                       ZMAI0057
    2 IF(IKICK.NE.0.AND.IMODE.NE.0) GO TO 7                             ZMAI0058
C                                                                       ZMAI0059
C     INTEGRATE VERTICAL RISE AND INITIALIZE IMODE                      ZMAI0060
      IMODE = 0                                                         ZMAI0061
      CALL MAINB                                                        ZMAI0062
C                                                                       ZMAI0063
C     CHOOSE THREE GRID KICK ANGLES CLOSEST TO TKICK                    ZMAI0064
    3 TK=FLOAT(IFIX((TKICK+.05)*10.0))/10.0                             ZMAI0065
    4 IF(ABS(TK-TKICK).LE.DELTK) GO TO 5                                ZMAI0066
      TK=TK+SIGN(DELTK,TKICK-TK)                                        ZMAI0067
      GO TO 4                                                           ZMAI0068
    5 NRUN=3                                                            ZMAI0069
      DO 6J = 1,NRUN                                                    ZMAI0070
    6 BRUN(J)=TK+FLOAT(J-2)*DELTK                                       ZMAI0071
C                                                                       ZMAI0072
      IF(IMODE.NE.0) GO TO17                                            ZMAI0073
      I = 3                                                             ZMAI0074
      GO TO 25                                                          ZMAI0075
C                                                                       ZMAI0076
C     THIS SECTION (THROUGH STATEMENT NUMBER 18) INTERROGATES THE LIST  ZMAI0077
C     OF KICK ANGLES FOR WHICH TRAJECTORIES HAVE BEEN RUN AND DETERMINESZMAI0078
C     WHICH OF THESE ARE SATISFACTORY FOR USE IN THE INTERPOLATION      ZMAI0079
C     SCHEME AND ON THE BASIS OF THOSE ALREADY AVAILABLE, CHOOSES NEW   ZMAI0080
C     KICK ANGLES TO BE RUN TO COMPLETE A SATISFACTORY SET OF THREE KICKZMAI0081
C     ANGLES FOR USE BY THE BOOST SUBROUTINE.                           ZMAI0082
C                                                                       ZMAI0083
C     FIND INTERVAL IN GRID INTO WHICH TKICK FALLS.                     ZMAI0084
    7 IF(TKICK.LT.APT(1,JKICK)) GO TO 8                                 ZMAI0085
      IF(JKICK.EQ.IKICK) GO TO 9                                        ZMAI0086
      JKICK=JKICK+1                                                     ZMAI0087
      GO TO 7                                                           ZMAI0088
    8 IF(JKICK.EQ.1.OR.TKICK.GE.APT(1,JKICK-1)) GO TO 9                 ZMAI0089
      JKICK=JKICK-1                                                     ZMAI0090
      GO TO 8                                                           ZMAI0091
C                                                                       ZMAI0092
C     FIND THREE CLOSEST GRID KICK ANGLES TO TKICK                      ZMAI0093
    9 JA= MAX0(1,JKICK-2)                                               ZMAI0094
      JB= MIN0(IKICK,JKICK+2)                                           ZMAI0095
      JC=JB-JA+1                                                        ZMAI0096
      DO 10 I=1,JC                                                      ZMAI0097
      JD=JA+I-1                                                         ZMAI0098
      RKA(I)=FLOAT(JD)                                                  ZMAI0099
   10 RK(I)=ABS(APT(1,JD)-TKICK)                                        ZMAI0100
      CALL SORTXY(RK,RKA,JC)                                            ZMAI0101
C                                                                       ZMAI0102
      KK = RKA(1)                                                       ZMAI0103
C                                                                       ZMAI0104
C     RKA(1) CONTAINS THE NUMBER OF THE CLOSEST KICK ANGLE.             ZMAI0105
C                                                                       ZMAI0106
C     DETERMINE KICK ANGLE TO BE USED FROM THE EXISTENT KICK ANGLES IN  ZMAI0107
C     THE GRID AND CHOOSE ANY NEW KICK ANGLES WHICH MUST BE INTEGRATED. ZMAI0108
      IF(RK(1).GE.2.0*DELTK) GO TO 3                                    ZMAI0109
      IF(RK(1).LT.DELTK) GO TO 12                                       ZMAI0110
      DELTKP=SIGN(DELTK,TKICK-APT(1,KK))                                ZMAI0111
      NRUN=2                                                            ZMAI0112
      DO 11 K = 1,2                                                     ZMAI0113
      KA=K                                                              ZMAI0114
```

```
      IF(DELTKP.LT.0.0) KA=3-K
   11 BRUN(KA)=APT(1,KK)+FLOAT(K)*DELTKP
      GO TO 17
C     PICK KICK ANGLES FROM EXISTING GRID FOR INTERPOLATION USE.
   12 IF(RK(3).LE.3.0*DELTK) GO TO 13
      IF(RK(2).GT.2.0*DELTK) GO TO 15
      NRUN=1
      BRUN(1)=APT(1,KK) + SIGN(DELTK,TKICK-APT(1,KK))
      GO TO 17
   13 DO 14 J = 1,3
      KA=RKA(J)
      APTMAX(J)=APT(1,KA)-TKICK
   14 ITK(J)=KPT(2,KA)
      GO TO 29
   15 NRUN=2
      DO 16 J = 1,2
   16 BRUN(J)=APT(1,KK)+(-1.0)**J*DELTK
   17 IF(BRUN(NRUN).LE.90.0-DELTK) GO TO 19
      NRUN=3
      TK=90.0-4.0*DELTK
      DO 18  JA=1,3
   18 BRUN(JA)=TK+FLOAT(JA)*DELTK
C
C     COMPARE LIST OF NEW KICK ANGLES TO BE RUN (BRUN(J)) WITH EXISTENT
C     KICK ANGLES AND ELIMINATE ANY DUPLICATION.
   19 I = NRUN
      DO 24 J = 1,IKICK
      K = IKICK-J+1
      KA=K+I
   20 IF(ABS(BRUN(I)-APT(1,K)).GT.0.000001*DELTK) GO TO 22
      NRUN=NRUN-1
      IF(I.EQ.NRUN+1) GO TO 19
      IF(NRUN.EQ.0) GO TO 13
      DO 21  JA=1,NRUN
   21 BRUN(JA)=BRUN(JA+1)
      GO TO 19
C
C     ALSO PUT NEW KICK ANGLES IN PROPER POSITION (ASCENDING ORDER)
C     IN APT(1,J).
   22 IF (BRUN(I).LT.APT(1,K)) GO TO 23
      APT(1,KA) = BRUN(I)
      KPT(2,KA) = IKICK+I
      I = I-1
      IF(I.EQ.0) GO TO 27
      KA=KA-1
      GO TO 20
   23 APT(1,KA) = APT(1,K)
      KPT(2,KA)=KPT(2,K)
C
   24 CONTINUE
   25 DO 26 J = 1,I
      APT(1,J)=BRUN(J)
   26 KPT(2,J)=IKICK+J
C
C
C     INTEGRATION OF NEW KICK ANGLES
C
C     IMODE EQUAL TO ONE FOR INTEGRATING BOOSTER AFTER VERTICAL RISE
   27 IMODE=1
C
```

```
C     TBLAST EQUALS MAXIMUM TIME IN BOOST TABLE
      TBOOST(LAST1)=TBLAST
C
C     CONTROLS FOR PRINT OUT (SEE STEP)
      STEPS=1000
      TMIN=TMINST
      DELMAX=DELTB
C
C     PRINT HEADINGS
      CALL OUTPT2(0)
      DO 28 J = 1,NRUN
C
C     SET MODOUT FOR INTEGRATING TABLE TRAJECTORY
      MODOUT=1
C
      IBURN=0
      ELEV=BRUN(J)
      IF(MODEC.NE.1) WRITE (6,34) ELEV
C
C     INCREMENT IKICK BY ONE FOR EACH NEW TRAJECTORY
      IKICK=IKICK+1
C
C     INTEGRATION OF NEW KICK ANGLES CONTROLLED BY MAINB
      CALL MAINB
C
   28 KPT(3,IKICK)=IBURN
C
C     RESET LAST
      LAST=5
C
C     SET IMODE EQUAL TO TWO FOR UPPER STAGE OPERATION.
      IMODE=2
      GO TO 7
   29 KK=ITK(1)
C
C     DETERMINE BOOSTER BURNING TIMES TO BE USED IN INTERPOLATION
C     SCHEME.  ITB(J) S CORRESPOND TO THE TIMES OF BOOSTER BURNOUT
C     POINTS.
      TBURN=TB(1)+TSPM
      ITB(1)=IFIX((TBURN+DELTB/2.0-TMINST)/DELTB)+1
      IF(ITB(1).LT.1.OR.ITB(1).GT.KPT(3,KK)) GO TO 33
      ALOG=ITB(1).GT.1
      IF(ALOG.AND.ITB(1).LT.KPT(3,KK)) GO TO 31
      IF(ALOG) GO TO 30
      ITB(2)=KPT(3,KK)-1
      ITB(3)=ITB(2)-1
      GO TO 32
   30 ITB(2)=2
      ITB(3)=3
      GO TO 32
   31 ITB(2)=ITB(1)-IFIX(SIGN(1.0,FLOAT(ITB(1))*DELTB-TBURN+TMINST))
      ITB(3)=ITB(1)+ISIGN(1,ITB(1)-ITB(2))
C
C
C     BOOST PERFORMS INTERPOLATION.
   32 CALL BOOST(VAR,NVAR,NKICKS)
C
      RETURN
   33 MASH=1
      RETURN
   34 FORMAT(14H0KICK ANGLE = ,G14.7)
      END
```

```
      SUBROUTINE MAINB                                                  ZMAI0001
C                                                                       ZMAI0002
C          SUBROUTINE MAINB PERFORMS THE INITIALIZATION NECESSARY       ZMAI0003
C     FOR INTEGRATING THE BOOSTER PHASE INCLUDING THE VERTICAL          ZMAI0004
C     RISE PORTION.  THE TIME INTERVAL (DELTB) BETWEEN TABLE STORAGE    ZMAI0005
C     POINTS IS CALCULATED IN THIS ROUTINE.                             ZMAI0006
C                                                                       ZMAI0007
      COMMON /CSTAR/ CMA(1000),CMB(1000)                                ZMAI0008
      COMMON /ATABLE/ CME(8000)                                         ZMAI0009
      DIMENSION ANGLEB(4       ),ANGLES(4       ),HARDB (6       )      ZMAI0010
      DIMENSION PERB  (2       ),RB    (5       ),SINA  (4       )      ZMAI0011
      DIMENSION TBOOST(6       ),TS    (6       ),WTFLOW(6       )      ZMAI0012
      DIMENSION X     (100     ),XPRIM (100,2   ),XSTORO(4       )      ZMAI0013
      EQUIVALENCE (A     ,CME(004)),(ALT   ,CMF(005)),(ALTE  ,CMA(805)) ZMAI0014
      EQUIVALENCE (ANGLEB,CME(025)),(ANGLES,CMA(786)),(ASTART,CMA(798)) ZMAI0015
      EQUIVALENCE (B     ,CMA(814)),(DELT  ,CMA(701)),(DELTB ,CME(016)) ZMAI0016
      EQUIVALENCE (DELTBT,CMA(803)),(ELEV  ,CMA(790)),(ESTART,CMA(795)) ZMAI0017
      EQUIVALENCE (FUELT ,CME(014)),(HARDB ,CMA(721)),(HARDBT,CME(015)) ZMAI0018
      EQUIVALENCE (HSTORE,CME(012)),(IKCKST,CMB(066)),(IKICK ,CME(200)) ZMAI0019
      EQUIVALENCE (IMODE ,CMB(061)),(JCOST ,CMB(129)),(JKICK ,CMB(063)) ZMAI0020
      EQUIVALENCE (LAST  ,CMA(711)),(LAST1 ,CMA(753)),(NKICK ,CME(020)) ZMAI0021
      EQUIVALENCE (NKICKS,CME(021)),(NSTAGE,CMA(710)),(NVAR  ,CMB(073)) ZMAI0022
      EQUIVALENCE (OBLATN,CME(002)),(OBLATS,CMA(815)),(PERB  ,CMB(055)) ZMAI0023
      EQUIVALENCE (RADIUS,CME(013)),(RB    ,CMA(754)),(REVOLV,CMA(799)) ZMAI0024
      EQUIVALENCE (SINA  ,CMA(791)),(STOA  ,CMA(813)),(TBLAST,CME(018)) ZMAI0025
      EQUIVALENCE (TBO   ,CME(017)),(TBOOST,CMA(745)),(TIME  ,CMA(017)) ZMAI0026
      EQUIVALENCE (TKTIME,CMA(804)),(TMINST,CME(019)),(TS    ,CMA(932)) ZMAI0027
      EQUIVALENCE (TSPM  ,CME(010)),(TSTART,CMA(796)),(VAT5  ,CMA(768)) ZMAI0028
      EQUIVALENCE (VEL   ,CME(001)),(VSTART,CMA(797)),(WSTORE,CMF(011)) ZMAI0029
      EQUIVALENCE (WTFLOW,CMA(733)),(WTO   ,CMA(720)),(X     ,CMA(401)) ZMAI0030
      EQUIVALENCE (XPRIM ,CMA(001)),(XSTORO,CME(006))                   ZMAI0031
      DOUBLE PRECISION XPRIM,TIME                                       ZMAI0032
      EXTERNAL EQUAT2,OUTPT2                                            ZMAI0033
      DATA RADDEG/57.2957795/                                           ZMAI0034
      SINA(1)=SIN(ANGLES(1)/RADDEG)                                     ZMAI0035
      NSTAGE=1                                                          ZMAI0036
C                                                                       ZMAI0037
C     SET JCOST TO BYPASS COAST ROUTINE                                 ZMAI0038
      JCOST = 1                                                         ZMAI0039
C                                                                       ZMAI0040
C                                                                       ZMAI0041
      IF(IMODE.NE.0) GO TO 7                                            ZMAI0042
C     IMODE EQUAL TO ZERO IMPLIES THAT A VERTICAL RISE IS TO BE         ZMAI0043
C     INTEGRATED.  IMODE NOT EQUAL TO ZERO IMPLIES THAT THE ZERO ANGLE  ZMAI0044
C     OF ATTACK PORTION IS TO BE INTEGRATED.                            ZMAI0045
C                                                                       ZMAI0046
C     INITIALIZE IKICK AND JKICK                                        ZMAI0047
      IKICK=0                                                           ZMAI0048
      JKICK=1                                                           ZMAI0049
C                                                                       ZMAI0050
      NKICK=7500/25/NVAR                                                ZMAI0051
      NKICKS=25*NKICK                                                   ZMAI0052
      FUELT=0.0                                                         ZMAI0053
      TSPM=TSTART                                                       ZMAI0054
```

```
      HARDBT=0.0
      IKCKST = 0
C
C     TBO EQUALS WEIGHT AT BEGINNING OF TABLE STORAGE.
      TBO=PERB(1)*WTO
C
C     LAST1 GREATER THAN ONE IMPLIES THAT A MULTI-SEGMENT BOOSTER IS
C     DESIRED.
      IF(LAST1.NE.1) GO TO 1
C
C
C     TBLAST EQUALS TIME AT END OF TABLE STORAGE.
      TBLAST=PERB(2)*WTO/WTFLOW(1)+TSTART
C
C     TMINST EQUALS TIME AT BEGINNING OF TABLE STORAGE.
      TMINST=TBO/WTFLOW(1)+TSTART
C
C     DELTB EQUALS TIME INTERVAL BETWEEN STORAGES.
      DELTB=(TBLAST-TMINST)/24.0
C
      GO TO 3
C
C     THE BEGINNING, END, AND INTERVAL OF TABLE STORAGE ARE CALCULATED
C     FOR A MULTI-SEGMENT BOOSTER.
    1 LAST1A=LAST1-1
      DO 2 J = 1,LAST1A
      TSPM=TSPM+TBOOST(J)
      HARDBT=HARDBT+HARDB(J)
    2 FUELT=FUELT+WTFLOW(J)*TBOOST(J)
      TMINST=(TBO-FUELT-HARDBT)/WTFLOW(LAST1)+TSPM
      TBLAST=(PERB(2)*WTO-FUELT-HARDBT)/WTFLOW(LAST1)
      DELTB=(TSPM+TBLAST-TMINST)/24.0
C
C     THIS SECTION PREPARES THE INITIAL STATE CONDITIONS WHICH ARE
C     CONVERTED INTO RECTANGULAR COORDINATES IN TUDES.
C     ESTART    ELEVATION ANGLE AND EQUALS 90 DEGREES
C     TSTART    INITIAL TIME
C     ASTART    INITIAL ALTITUDE
C     OBLATN    ONE IF OBLATENESS IS USED, ZERO IF NOT
    3 ANGLES(4)=ESTART
      TIME=DBLE(TSTART)
      VEL=VSTART
      ALT=ASTART
      OBLATN=OBLATS
      A = STOA
      IF(OBLATN.NE.0.0) GO TO 4
      A=A*B/SQRT(B**2+(A**2-B**2)*SINA(1)**2)
      RADIUS=A+ALT
      GO TO 5
    4 RADIUS=A*B/SQRT(B**2+(A**2-B**2)*SINA(1)**2)+ALT
    5 CALL TUDES
C
C     XSTORO STORES THE INITIAL X, Y, Z COMPONENTS OF RADIUS FOR USE
C     IN CALCULATING TRAVEL ANGLE, ETC.
      XSTORO(1)=SNGL(XPRIM(6,1))
      XSTORO(2)=SNGL(XPRIM(7,1))
      XSTORO(3)=SNGL(XPRIM(8,1))
      XSTORO(4)=DOT(XSTORO,XSTORO)
C
      LAST=1
```

```
C
C     TKTIME IS THE LENGTH OF VERTICAL RISE
      TS(1)=TKTIME+TSTART
C
C
C     HEAT INTEGRAL POSITION IN XPRIM (SEE RUNGEK) IS ZEROED.
      XPRIM(1,1)=0.0D0
C
C
C     WTO IS THE INITIAL GROSS WEIGHT.
      XPRIM(2,1)=DBLE(WTO)
C
C
C     OUTPT2(0) PRINTS HEADINGS
      CALL OUTPT2(0)
C
C     SETUP PREPARES THE CONTROL PARAMETERS (MODOUT, MODS, ETC.) FOR
C     STEP.
      CALL SETUP
C
C
C     RUNGEK INTEGRATES THE REQUIRED TRAJECTORY.
      CALL RUNGEK (EQUAT2,OUTPT2)
C
C     X(1) IS THE HEAT INTEGRAL AT THE END OF THE VERTICAL RISE.
      HSTORE=X(1)
C
C     WSTORE STORES THE WEIGHT AT THE END OF THE VERTICAL RISE.
      WSTORE=X(2)
C
C     ANGLEB STORES THE INERTIAL LATITUDE, LONGITUDE, AND AZIMUTH
C     AT THE END OF THE VERTICAL RISE.
      ANGLEB(1)=ARSIN(RB(3)/RB(5))*RADDEG
      ANGLEB(2)=ARCTAN(RB(2),RB(1))*RADDEG
      ANGLEB(3)=ANGLES(3)
C
C     ALT IS A STORAGE FOR ALTITUDE AT THE END OF THE VERTICAL RISE
      ALT=ALTE
C
C
C     VEL IS A STORAGE FOR VELOCITY AT THE END OF THE VERTICAL RISE
      VEL=VAT5
C
C     RADIUS IS A STORAGE FOR THE RADIUS AT THE END OF THE VERTICAL
C     RISE.  IT IS COMPUTED DIFFERENTLY DEPENDING ON WHETHER THE
C     OBLATE EARTH MODEL IS UTILIZED.
      IF(OBLATN.EQ.0.0) GO TO 6
      RADIUS=A*B/SQRT(B**2+(A**2-B**2)*SINA(1)**2)+ALT
      RETURN
    6 RADIUS=A+ALT
C
      RETURN
C
C     THE REMAINING PORTION OF THIS ROUTINE CONTROLS THE INTEGRATION
C     OF THE PORTION OF THE BOOSTER AFTER THE VERTICAL RISE.
C
C     ELEV EQUALS THE KICK ANGLE AND IS SET IN MAINA
    7 ANGLEB(4)=ELEV
C
C     TUDES CONVERTS LATITUDE, LONGITUDE, AZIMUTH, ELEVATION ANGLE,
```

```
C     AND VELOCITY TO RECTANGULAR COORDINATES.                        ZMAI0175
      CALL TUDES                                                      ZMAI0176
C                                                                     ZMAI0177
      TIME=DBLE(TKTIME+TSTART)                                        ZMAI0178
      LAST=LAST1                                                      ZMAI0179
C                                                                     ZMAI0180
C     SET HEAT TO VALUE AT END OF VERTICAL RISE.                      ZMAI0181
      XPRIM(1,1)=DBLE(HSTORE)                                         ZMAI0182
C                                                                     ZMAI0183
C                                                                     ZMAI0184
C     SET WEIGHT TO WEIGHT AT END OF VERTICAL RISE.                   ZMAI0185
      XPRIM(2,1)=DBLE(WSTORE)                                         ZMAI0186
C                                                                     ZMAI0187
C                                                                     ZMAI0188
C     THE TS ARRAY CONTAINS THE TIMES AT WHICH PHASE CHANGES OCCUR    ZMAI0189
C     DURING THE BOOSTER.                                             ZMAI0190
      TS(1)=TSTART+TBOOST(1)                                          ZMAI0191
      IF (LAST1.EQ.1) GO TO 9                                         ZMAI0192
      DO 8 I = 2,LAST1                                                ZMAI0193
    8 TS(I) = TS(I-1)+TBOOST(I)                                       ZMAI0194
C                                                                     ZMAI0195
    9 DELT=AMIN1(DELTBT,TS(1)-TIME)                                   ZMAI0196
C                                                                     ZMAI0197
C     RUNGEK INTEGRATES THE TRAJECTORY.                               ZMAI0198
      CALL RUNGEK (EQUAT2,OUTPT2)                                     ZMAI0199
C                                                                     ZMAI0200
      RETURN                                                          ZMAI0201
      END                                                             ZMAI0202
```

```
      SUBROUTINE OBLATE                                                 ZOBL0001
C         THIS SUBROUTINE COMPUTES THE OBLATENESS ACCELERATIONS         ZOBL0002
C     (OBLAT) DUE TO AN AXIALLY SYMMETRIC EARTH.  THE 2ND, 3RD, AND 4TH ZOBL0003
C     SPHERICAL HARMONIC COEFFICIENTS ARE OBLATJ, OBLATH, AND OBLATD    ZOBL0004
C     RESPECTIVELY.   THESE CONSTANTS ARE LOADED IN DATAB.              ZOBL0005
      COMMON /CSTAR/ CMA(1000),CMB(1000)                                ZOBL0006
      DIMENSION OBLAT (3      ),RB    (5      )                         ZOBL0007
      EQUIVALENCE (A     ,CMA(813)),(GM    ,CMA(715)),(OBLAT ,CMA(780)) ZOBL0008
      EQUIVALENCE (OBLATD,CMA(816)),(OBLATH,CMA(817)),(OBLATJ,CMA(818)) ZOBL0009
      EQUIVALENCE (RB    ,CMA(754))                                     ZOBL0010
C                                                                       ZOBL0011
C                                                                       ZOBL0012
C                                                                       ZOBL0013
C                                                                       ZOBL0014
      AA = RB(3)/RB(5)                                                  ZOBL0015
      AB = RB(3)**2/RB(4)                                               ZOBL0016
      AC = A **2/RB(4)                                                  ZOBL0017
      AD = GM/RB(4)/RB(5)*AC                                            ZOBL0018
      AE = OBLATJ*AD                                                    ZOBL0019
      AF = OBLATH*AD* A/RB(5)                                           ZOBL0020
      AG = OBLATD*AD*AC                                                 ZOBL0021
      AH = AE*(5.0*AB-1.0)+AF*(7.0*AB-3.0)*AA+AG*(6.0*AB-9.0*AB**2-.4285ZOBL0022
     171428E+00)                                                        ZOBL0023
      OBLAT(1) = AH*RB(1)                                               ZOBL0024
      OBLAT(2) = AH*RB(2)                                               ZOBL0025
      OBLAT(3)=(AH-2.0*AE+AG*(4.0*AB-.171428571E+01))*RB(3)-AF*(3.0*AB-0ZOBL0026
     1.6)*RB(5)                                                         ZOBL0027
    1 RETURN                                                            ZOBL0028
      END                                                               ZOBL0029
```

```
C      FUNCTION ODDMOD IS SIMPLY AN IMPROVEMENT OF                      ZODM0001
C      LIBRARY MOD BY MAKING ALGEBRAIC SIGN OF                          ZODM0002
C      ARGUMENT 1 AND ARGUMENT 2 IDENTICAL BEFORE                       ZODM0003
C      PERFORMING MOD.                                                  ZODM0004
C                                                                       ZODM0005
      FUNCTION ODDMOD(A1,A2)                                            ZODM0006
  100 A1=A1+A2                                                          ZODM0007
      IF(A1/A2)100,101,101                                              ZODM0008
  101 ODDMOD=AMOD(A1,A2)                                                ZODM0009
      RETURN                                                            ZODM0010
      END                                                               ZODM0011
```

```
      SUBROUTINE ORBEL
C
C          SUBROUTINE ORBEL COMPUTES ORBIT ELEMENTS FROM THE POLAR
C     COORDINATES IN WHICH THE UPPER PHASES ARE INTEGRATED.
C
      COMMON /CSTAR/ CMA(1000),CMB(1000)
      EQUIVALENCE (E      ,CMA(905)),(FM     ,CMA(715)),(OMEGA  ,CMA(405))
      EQUIVALENCE (P      ,CMA(906)),(PERIOD,CMA(910)),(PHI    ,CMA(403))
      EQUIVALENCE (R      ,CMA(402)),(THETA  ,CMA(909)),(TIME   ,CMA(409))
      EQUIVALENCE (TP     ,CMA(907)),(TPD    ,CMA(908)),(U      ,CMA(404))
      EQUIVALENCE (V      ,CMA(889))
C
C     DEFINITIONS AND UNITS FOR VARIABLES APPEARING IN THIS ROUTINE.
C     FM         GRAVITATIONAL CONSTANT FOR THE EARTH        FT**3/SEC**2
C     OMEGA      ANGULAR VELOCITY                            RAD/SEC
C     E          ECCENTRICITY
C     P          SEMILATUS RECTUM                            FT
C     PERIOD     PERIOD OF THE ELLIPSE                       SEC
C     PHI        TRAVEL ANGLE                                RAD
C     THETA      TRUE ANOMALY                                DEG
C     TP         TIME OF PERIGEE                             SEC
C     TPD        TIME OF PERIGEE DEPARTURE                   SEC
C     V          VELOCITY                                    FT/SEC
C     U          RADIAL VELOCITY                             FT/SEC
C     R          RADIUS, MEASURED FROM THE CENTER OF         FT
C                THE EARTH
C
      RSQ = R*R
C
C     CALCULATE SEMILATUS RECTUM
      P = (RSQ*OMEGA)*(RSQ*OMEGA)/FM
C
      VSQ = U*U+RSQ*OMEGA*OMEGA
      V = SQRT(VSQ)
      IF(U.NE.0.0) GO TO 2
C
C     ORBIT ELEMENTS FOR CIRCULAR ORBIT
      TP = 0.0
      E=0.0
      TPD = 0.0
      SINT = 0.0
      COST = 1.0
      THETA = 0.0
C
C     CALCULATE PERIOD OF CIRCULAR ORBIT
    1 PERIOD = 6.2831853*R*SQRT(R/FM)
C
      RETURN
C
C     CALCULATE ECCENTRICITY
    2 E = SQRT(1.0-P*(2.0/R-VSQ/FM))
C
      IF(E.EQ.0.0) GO TO 1
C
C     CALCULATE TRUE ANOMALY
      COST = (P/R-1.0)/E
```

```
      SINT = U*SQRT(P/FM)/E                                             ZORB0057
      THETA = ARCTAN(SINT,COST)*57.2957795                              ZORB0058
C                                                                       ZORB0059
C     CALCULATE TIME OF PERIGEE DEPARTURE                               ZORB0060
      ECOST = 1.0+E*COST                                                ZORB0061
      EPAR1 = ABS(1.0-E*E)                                              ZORB0062
      EPAR2 = SQRT(EPAR1)                                               ZORB0063
      IF(COST.NE.(-1.0))GO TO 3                                         ZORB0064
      TPDPAR = .314159265E+01/(EPAR1*EPAR2)                             ZORB0065
      GO TO 7                                                           ZORB0066
    3 TANPAR = SINT/(1.0+COST)                                          ZORB0067
      IF(E-1.0) 6,4,5                                                   ZORB0068
    4 TPDPAR = (ECOST+1.0)*TANPAR/(3.0*ECOST)                           ZORB0069
      GO TO 7                                                           ZORB0070
    5 TPDPAR= E*SINT/(EPAR1*ECOST)-ALOG((E+1.0+EPAR2*TANPAR)/(E+1.0-   EZORB0071
     1PAR2*TANPAR))/(EPAR1*EPAR2)                                       ZORB0072
      GO TO 7                                                           ZORB0073
    6 TPDPAR = -E*SINT/(EPAR1*ECOST)+2.0*ATAN(EPAR2*TANPAR/(1.0+E))/   (ZORB0074
     1EPAR1*EPAR2)                                                      ZORB0075
C                                                                       ZORB0076
C     CALCULATE PERIOD OF ELLIPSE                                       ZORB0077
C                                                                       ZORB0078
      PERIOD = 6.2831853*P*SQRT(P/FM)/(EPAR1*EPAR2)                     ZORB0079
    7 TPD = TPDPAR*P*SQRT(P/FM)                                         ZORB0080
C                                                                       ZORB0081
C     CALCULATE TIME OF PERIGEE                                         ZORB0082
      TP = TIME-TPD                                                     ZORB0083
C                                                                       ZORB0084
      RETURN                                                            ZORB0085
C     END OF FORTRAN STATEMENTS                                         ZORB0086
      END                                                               ZORB0087
```

```
      SUBROUTINE ORBEL2                                                 ZORB0001
C                                                                       ZORB0002
C          SUBROUTINE ORBEL2 CALCULATES ORBIT ELEMENTS AND RELATED      ZORB0003
C     INFORMATION FROM THE RECTANGULAR COORDINATES IN WHICH THE         ZORB0004
C     BOOSTER IS INTEGRATED.                                            ZORB0005
C                                                                       ZORB0006
      COMMON /ATABLE/ CME(8000)                                         ZORB0007
      COMMON /CSTAR/ CMA(1000),CMB(1000)                                ZORB0008
      DIMENSION H      (5       ),HATM   (5       ),RB     (5       )   ZORB0009
      DIMENSION RIC    (5       ),VATM   (5       ),VX     (5       )   ZORB0010
      DIMENSION XSTORE(4       ),XSTORO(4       )                       ZORB0011
      EQUIVALENCE (ALTE  ,CMA(805)),(AZIMI ,CMA(921)),(AZIMR ,CMA(922)) ZORB0012
      EQUIVALENCE (BETAI ,CMA(917)),(BETAR ,CMA(918)),(E     ,CMA(905)) ZORB0013
      EQUIVALENCE (FM    ,CMA(715)),(H     ,CMA(769)),(P     ,CMA(906)) ZORB0014
      EQUIVALENCE (PERIOD,CMA(910)),(PHII  ,CMA(911)),(PHIR  ,CMA(912)) ZORB0015
      EQUIVALENCE (RANGE ,CMA(914)),(RB    ,CMA(754)),(RDOT  ,CMB(155)) ZORB0016
      EQUIVALENCE (RE    ,CMA(924)),(REVOLV,CMA(799)),(THETA ,CMA(909)) ZORB0017
      EQUIVALENCE (TIME  ,CMA(409)),(TP    ,CMA(907)),(TPD   ,CMA(908)) ZORB0018
      EQUIVALENCE (TRAVLI,CMA(913)),(VATM  ,CMA(764)),(VX    ,CMA(759)) ZORB0019
      EQUIVALENCE (X     ,CMA(401)),(XSTORO,CME(006)),(ZINCLI,CMA(919)) ZORB0020
      EQUIVALENCE (ZINCLR,CMA(920)),(ZLAT  ,CMA(916)),(ZLONG ,CMA(915)) ZORB0021
      EQUIVALENCE (ZNODEI,CMA(923))                                     ZORB0022
C                                                                       ZORB0023
C     DEFINITIONS AND UNITS FOR VARIABLES APPEARING IN THIS ROUTINE     ZORB0024
C     ALTE       ALTITUDE ABOVE OBLATE EARTH                FT          ZORB0025
C     AZIMI      INERTIAL AZIMUTH HEADING                   DEG         ZORB0026
C     AZIMR      RELATIVE AZIMUTH HEADING                   DEG         ZORB0027
C     BETAR      RELATIVE FLIGHT PATH ANGLE                 DEG         ZORB0028
C     E          ECCENTRICITY                                           ZORB0029
C     FM         GRAVITATIONAL CONSTANT FOR EARTH           FT**3/SEC**2 ZORB0030
C     H          ANGULAR MOMENTUM VECTOR                    FT**2/SEC   ZORB0031
C     PERIOD     PERIOD OF ELLIPSE                          SEC         ZORB0032
C     PHIR       RELATIVE TRAVEL ANGLE                      DEG         ZORB0033
C     PHII       INERTIAL TRAVEL ANGLE                      DEG         ZORB0034
C     P          SEMILATUS RECTUM                           FT          ZORB0035
C     RANGE      DISTANCE TO LAUNCH SITE ASSUMING           NM          ZORB0036
C                SPHERICAL EARTH WITH RADIUS EQUAL TO                   ZORB0037
C                RADIUS AT INSTANTANEOUS POSITION OF                    ZORB0038
C                THE VEHICLE                                            ZORB0039
C     RB         POSITION VECTOR                            FT          ZORB0040
C     RDOT       RADIAL VELOCITY                            FT/SEC      ZORB0041
C     REVOLV     ANGULAR VELOCITY OF THE EARTH              RAD/SEC     ZORB0042
C     RE         INSTANTANEOUS RADIUS OF THE OBLATE EARTH   FT          ZORB0043
C     THETA      TRUE ANOMALY                               DEG         ZORB0044
C     TPD        TIME OF PERIGEE DEPARTURE                  SEC         ZORB0045
C     TP         TIME OF PERIGEE                            SEC         ZORB0046
C     TRAVLI     TOTAL INERTIAL DISTANCE TRAVELED OVER      NM          ZORB0047
C                A SPHERICAL EARTH WITH RADIUS EQUAL TO THE             ZORB0048
C                RADIUS OF THE EARTH AT THE INSTANTANEOUS               ZORB0049
C                POSITION.                                              ZORB0050
C     VX         INERTIAL VELOCITY VECTOR                   FT/SEC      ZORB0051
C     XSTORO     POSITION VECTOR OF LAUNCH SITE             FT          ZORB0052
C     ZINCLI     INERTIAL INCLINATION                       DEG         ZORB0053
C     ZINCLR     RELATIVE INCLINATION                       DEG         ZORB0054
C     ZLAT       LATITUDE                                   DEG         ZORB0055
C     ZLONG      LONGITUDE                                  DEG         ZORB0056
```

```
C     ZNODEI    ASCENDING NODE                              DEG         ZORB0057
C     VATM      RELATIVE VELOCITY VECTOR                    FT/SEC      ZORB0058
C                                                                       ZORB0059
      DATA RADDEG /57.2957795/                                          ZORB0060
C                                                                       ZORB0061
C     CALCULATE SEMILATUS RECTUM                                        ZORB0062
      P=H(4)/FM                                                         ZORB0063
C                                                                       ZORB0064
      IF(RDOT.NE.0.0) GO TO 1                                           ZORB0065
C                                                                       ZORB0066
C     ORBIT ELEMENTS FOR CIRCULAR ORBIT                                 ZORB0067
      TP = 0.0                                                          ZORB0068
      SINT = 0.0                                                        ZORB0069
      COST = 1.0                                                        ZORB0070
      E=0.0                                                             ZORB0071
      TPD = 0.0                                                         ZORB0072
      THETA = 0.0                                                       ZORB0073
C                                                                       ZORB0074
C     CALCULATE PERIOD OF CIRCULAR ORBIT                                ZORB0075
      PERIOD=6.2831853*RB(5)*SQRT(RB(5)/FM)                             ZORB0076
C                                                                       ZORB0077
      GO TO 8                                                           ZORB0078
C                                                                       ZORB0079
C     CALCULATE ECCENTRICITY                                            ZORB0080
    1 E=SQRT(ABS(1.0+P*(VX(4)/FM-2.0/RB(5))))                           ZORB0081
C                                                                       ZORB0082
      IF(E.NE.0.0) GO TO 2                                              ZORB0083
      PERIOD=6.2831853*P*SQRT(P/FM)                                     ZORB0084
      GO TO 8                                                           ZORB0085
C                                                                       ZORB0086
C     CALCULATE TRUE ANOMALY                                            ZORB0087
    2 COST = (P/RB(5)-1.0)/E                                            ZORB0088
      SINT = RDOT*SQRT(P/FM)/E                                          ZORB0089
      THETA =ARCTAN(SINT,COST)*RADDEG                                   ZORB0090
C                                                                       ZORB0091
C                                                                       ZORB0092
C                                                                       ZORB0093
C     CALCULATE TIME OF PERIGEE DEPARTURE                               ZORB0094
      ECOST = 1.0+E*COST                                                ZORB0095
      EPAR1 = ABS(1.0-E*E)                                              ZORB0096
      EPAR2 = SQRT(EPAR1)                                               ZORB0097
      IF(COST.NE.(-1.0)) GO TO 3                                        ZORB0098
      IF(EPAR1.EQ.0.0) GO TO 8                                          ZORB0099
      TPDPAR = .314159265E+01/(EPAR1*EPAR2)                             ZORB0100
      GO TO 7                                                           ZORB0101
    3 TANPAR = SINT/(1.0+COST)                                          ZORB0102
      IF(E-1.0) 6,4,5                                                   ZORB0103
    4 TPDPAR = (ECOST+1.0)*TANPAR/(3.0*ECOST)                           ZORB0104
      GO TO 7                                                           ZORB0105
    5 TPDPAR= E*SINT/(EPAR1*ECOST)-ALOG((E+1.0+EPAR2*TANPAR)/(E+1.0-   EZORB0106
     1PAR2*TANPAR))/(EPAR1*EPAR2)                                       ZORB0107
      GO TO 7                                                           ZORB0108
    6 TPDPAR = -E*SINT/(EPAR1*ECOST)+2.0*ATAN(EPAR2*TANPAR/(1.0+E))/   (ZORB0109
     1EPAR1*EPAR2)                                                      ZORB0110
C                                                                       ZORB0111
C     CALCULATE PERIOD OF ELLIPSE                                       ZORB0112
      PERIOD = 6.2831853*P*SQRT(P/FM)/(EPAR1*EPAR2)                     ZORB0113
C                                                                       ZORB0114
    7 TPD = TPDPAR*P*SQRT(P/FM)                                         ZORB0115
C                                                                       ZORB0116
```

```
      TP = TIME-TPD
C
    8 COSINC = H(3)/H(5)
      SINLAT = RB(3)/RB(5)
      COSLAT = SQRT(1.0-SINLAT**2)
      RXYSQ=RB(1)**2+RB(2)**2
      RXYRT=SQRT(RXYSQ)
      SINLON = RB(2)/RXYRT
      COSLON = RB(1)/RXYRT
      HROOT=SQRT(H(1)**2+H(2)**2)
      SINNOD = H(1)/HROOT
      COSNOD=-H(2)/HROOT
      COSPHI=COSLAT*(COSLON*COSNOD+SINLON*SINNOD)
C
C     CALCULATE ASCENDING NODE
      ZNODEI=ODDMOD(ARCTAN(H(1),-H(2))*RADDEG,360.0)
C
C     CALCULATE INERTIAL INCLINATION
      ZINCLI = ARCOS(COSINC)*RADDEG
C
C     CALCULATE LONGITUDE
      ZLONG=ODDMOD((ARCTAN(RB(2),RB(1))-REVOLV*TIME)*RADDEG,360.0)
C
C     CALCULATE LATITUDE
      ZLAT = ARSIN(SINLAT)*RADDEG
C
C     CALCULATE INERTIAL AZIMUTH
      AZIMI=ODDMOD(ARCTAN(H(3)/HROOT,COSPHI)*RADDEG,360.0)
C
      CALL CONVT(RB(1),VATM(1),HATM(1))
      HROOT=SQRT(HATM(1)**2+HATM(2)**2)
C
C     CALCULATE RELATIVE FLIGHT PATH ANGLE
      BETAR  = ARCTAN(RDOT,HATM(5)/RB(5))*RADDEG
C
C     IF RELATIVE ANGULAR MOMENTUM EQUALS ZERO, SET RELATIVE
C     INCLINATION AND AZIMUTH TO ZERO.
      IF(HATM(5).NE.0.0) GO TO 9
      ZINCLR=0.0
      AZIMR=0.0
      GO TO 10
C
C     CALCULATE RELATIVE INCLINATION
    9 COSINC = HATM(3)/HATM(5)
      ZINCLR = ARCOS(COSINC)*RADDEG
C
      IF(HROOT.EQ.0.0) GO TO 10
      SINNOD=HATM(1)/HROOT
      COSNOD=-HATM(2)/HROOT
      COSPHI=COSLAT*(COSLON*COSNOD+SINLON*SINNOD)
C
C     CALCULATE RELATIVE AZIMUTH ANGLE
      AZIMR=ODDMOD(ARCTAN(HATM(3)/HROOT,COSPHI)*RADDEG    ,360.0)
   10 SLIP = REVOLV*TIME
      COSLIP = COS(SLIP)
      SINLIP = SIN(SLIP)
      RIC(1)=XSTORO(1)*COSLIP-XSTORO(2)*SINLIP
      RIC(2)=XSTORO(2)*COSLIP+XSTORO(1)*SINLIP
      RIC(3)=XSTORO(3)
      CALL CONVT(RIC,RB,XSTORE)
```

```
C
C     CALCULATE INSTANTANEOUS RADIUS OF THE EARTH
      RE=RB(5)-ALTE
C
C     CALCULATE THE RELATIVE TRAVEL ANGLE
      PHIR=ABS(ARSIN(XSTORE(5)/XSTORO(4)/RB(5)))
C
C
C     CALCULATE RELATIVE DISTANCE TRAVERSED
      RANGE=RE*PHIR/6076.1155
C
      PHIR=PHIR*RADDEG
      CALL CONVT(XSTORO,RB,XSTORE)
C     CALCULATE THE INERTIAL TRAVEL ANGLE
      PHII=ABS(ARSIN(XSTORE(5)/XSTORO(4)/RB(5)))
C
C     CALCULATE INERTIAL DISTANCE TRAVERSED
      TRAVLI=RE*PHII/6076.1155
C
      PHII=PHII*RADDEG
C
      RETURN
      END
```

```
      SUBROUTINE OUTPT1(N)
C
C          SUBROUTINE OUTPT1 AND OUTPT2 PRINT THE DATA AND CONTAIN
C     MOST OF THE WRITE STATEMENTS IN THE PROGRAM.  OUTPT1 HANDLES
C     THE OUTPUT FOR THE UPPER PHASE PORTION OF THE PROGRAM.  IT IS
C     DIVIDED INTO TWO SECTIONS, ONE WHICH PRINTS HEADINGS AND THE
C     OTHER WHICH PRINTS THE NUMERICAL DATA
C
      COMMON /CSTAR/ CMA(1000),CMB(1000)
      COMMON /RUNG/RUN(125)
      DIMENSION AMF   (6       ),FLOMX (6       ),FUEL  (6       )
      DIMENSION HARD  (6       ),HARD1 (6       ),PROP  (6       )
      DIMENSION TB    (6       ),THRUST(6       ),TS    (6       )
      DIMENSION VELEXP(6       )
      EQUIVALENCE (CAPPA ,CMA(891)),(DELTAV,CMA(861)),(DROP   ,CMA(863))
      EQUIVALENCE (DZLAM1,CMA(506)),(DZLAM2,CMA(507)),(F      ,CMA(905))
      EQUIVALENCE (FDM   ,CMA(886)),(FLOMX ,CMA(837)),(FLOW   ,CMA(877))
      EQUIVALENCE (FUEL  ,CMA(871)),(FUELDV,CMA(878)),(G      ,CMA(716))
      EQUIVALENCE (HARD  ,CMA(843)),(IMODE ,CMB(061)),(LAST   ,CMA(890))
      EQUIVALENCE (NSTAGE,CMA(710)),(NSTAG1,RUN(115)),(OMEGA  ,CMA(405))
      EQUIVALENCE (P     ,CMA(906)),(PERIOD,CMA(910)),(PHI    ,CMA(403))
      EQUIVALENCE (PROP  ,CMA(849)),(PSI   ,CMA(885)),(R      ,CMA(402))
      EQUIVALENCE (RESERV,CMA(862)),(RMASS ,CMA(401)),(RO     ,CMA(904))
      EQUIVALENCE (STEPNO,RUN(112)),(TB    ,CMA(825)),(THETA  ,CMA(909))
      EQUIVALENCE (THRUST,CMA(831)),(TIME  ,CMA(409)),(TKICK  ,CMB(059))
      EQUIVALENCE (TP    ,CMA(907)),(TPD   ,CMA(908)),(TS     ,CMA(932))
      EQUIVALENCE (U     ,CMA(404)),(V     ,CMA(889)),(VELEX  ,CMA(870))
      EQUIVALENCE (VELEXP,CMA(864)),(ZLAM1 ,CMA(406)),(ZLAM2  ,CMA(407))
      INTEGER STEPNO
      DATA CONRTD /57.2957795/
C
C     N EQUAL TO ZERO IMPLIES THAT HEADINGS ARE TO BE PRINTED, N
C     EQUAL TO ONE CALLS FOR NUMERICAL OUTPUT.
      IF(N.NE.0) GO TO 5
      DO 1 J = 1,6
      HARD1(J)=0.0
    1 AMF(J)=0.0
C
C     THESE HEADINGS IDENTIFY THE THRUST, FLOW, FIXED HARDWARE,
C     AND THE PROPELLANT SENSITIVE MASS FRACTION FOR EACH PHASE.
      WRITE (6,2)
    2 FORMAT (4H0A1 ,
     114HTHRUST ONE    ,2X,14HTHRUST FOUR   ,2X,14HFLOW ONE      ,2X,
     114HFLOW FOUR     ,2X,14HHARD ONE      ,2X,14HHARD FOUR     ,2X,
     114HPROP ONE      ,2X,14HPROP FOUR     /4H A2 ,
     114HTHRUST TWO    ,2X,14HTHRUST FIVE   ,2X,14HFLOW TWO      ,2X,
     114HFLOW FIVE     ,2X,14HHARD TWO      ,2X,14HHARD FIVE     ,2X,
     114HPROP TWO      ,2X,14HPROP FIVE     /4H A3 ,
     114HTHRUST THREE  ,2X,14HTHRUST SIX    ,2X,14HFLOW THREE    ,2X,
     114HFLOW SIX      ,2X,14HHARD THREE    ,2X,14HHARD SIX      ,2X,
     114HPROP THREE    ,2X,14HPROP SIX      )
C
C     THESE HEADINGS IDENTIFY THE OUTPUT WHICH IS DERIVED FROM
C     THE TRAJECTORY.  THE VARIABLES ARE
C     TIME            TIME FROM LAUNCH                          SEC
C     WEIGHT          INSTANTANEOUS WEIGHT                      LB
```

```
C      STEPNO            THE NUMBER OF GOOD INTEGRATION STEPS                   ZOTP0057
C                        TAKEN UP TO AND INCLUDING THE STEP                     ZOTP0058
C                        PRINTED                                                ZOTP0059
C      CAPPA             SEE COAST FOR USE OF CAPPA                             ZOTP0060
C      KICK ANGLE        BOOSTER KICK ANGLE                     DEG             ZOTP0061
C      RADIUS            DISTANCE TO THE CENTER OF EARTH        FT              ZOTP0062
C      TIME PER DEP      TIME OF PERIGEE DEPARTURE              SEC             ZOTP0063
C      SEM LAT REC       SEMILATUS RECTUM                       FT              ZOTP0064
C      TRUE ANOMALY      TRUE ANOMALY                           DEG             ZOTP0065
C      HOR. VELOCITY     HORIZONTAL VELOCITY                    FT/SEC          ZOTP0066
C      PSI               ANGLE BETWEEN THRUST VECTOR AND        DEG             ZOTP0067
C                        LOCAL HORIZONTAL, MEASURED POSITIVE                    ZOTP0068
C                        CLOCKWISE                                              ZOTP0069
C      ALTITUDE          ALTITUDE ABOVE SPHERICAL EARTH         FT              ZOTP0070
C      ECCENTRICITY      ECCENTRICITY OF CONIC                                  ZOTP0071
C      INR. GAMMA        INERTIAL FLIGHT PATH ANGLE             DEG             ZOTP0072
C      PERIOD            PERIOD OF ELLIPSE                      SEC             ZOTP0073
C      INR. VELOCITY     TOTAL INERTIAL VELOCITY                FT/SEC          ZOTP0074
C      FLOW              FLOW RATE                              LB/SEC          ZOTP0075
C      ACCELLERATION     THRUST-TO-WEIGHT                       G S             ZOTP0076
C      PSID              DERIVATIVE OF THRUST ANGLE WITH        DEG/SEC         ZOTP0077
C                        RESPECT TO TIME                                        ZOTP0078
C      PERIGEE ALT       ALTITUDE OF PERIGEE POINT              FT              ZOTP0079
C      TIME OF PERIGEE   TIME OF PERIGEE REFERENCED TO TIME     SEC             ZOTP0080
C                        OF PRINTOUT                                            ZOTP0081
C      OMEGA             ANGULAR VELOCITY                       DEG/SEC         ZOTP0082
C      TRAVEL ANGLE      INERTIAL TRAVEL ANGLE MEASURED FROM    DEG             ZOTP0083
C                        THE BEGINNING OF THE UPPER PHASE POR-                  ZOTP0084
C                        TION OF THE PROBLEM                                    ZOTP0085
C      RAD VELOCITY      RADIAL VELOCITY                        FT/SEC          ZOTP0086
       WRITE (6,3)                                                              ZOTP0087
     3 FORMAT (4HOU1 ,                                                          ZOTP0088
      114HTIME          ,2X,14HWEIGHT        ,2X,14HPSI            ,2X,         ZOTP0089
      114HALTITUDE      ,2X,14HECCENTRICITY  ,2X,14HINR. GAMMA     ,2X,         ZOTP0090
      114HPERIOD        ,2X,14HINR. VELOCITYY/4H U2 ,                           ZOTP0091
      114HFLOW          ,2X,14HACCELLERATION ,2X,14HPSID           ,2X,         ZOTP0092
      114HPERIGEE ALT   ,2X,14HTIME OF PERIGE,2X,14HOMEGA          ,2X,         ZOTP0093
      114HTRAVEL ANGLE  ,2X,14HRAD. VELOCITY /4H U3 ,                           ZOTP0094
      114HSTEPNO        ,2X,14HCAPPA         ,2X,14HKICK ANGLE     ,2X,         ZOTP0095
      114HRADIUS        ,2X,14HTIME PER DEP  ,2X,14HSEM LAT REC    ,2X,         ZOTP0096
      114HTRUE ANOMALY  ,2X,14HHOR. VELOCITY )                                  ZOTP0097
C                                                                               ZOTP0098
C      THIS OUTPUT SECTION SUPPLIES INFORMATION WHICH COULD BE COMPUTED         ZOTP0099
C      FROM A TRAJECTORY, BUT FOR CONVENIENCE IS TABULATED AT THE               ZOTP0100
C      END OF A TRAJECTORY.  THE ITEMS LISTED FOR EACH PHASE ARE                ZOTP0101
```

```
C      BURNING TIME, PROPELLANT LOADING, TOTAL HARDWARE WEIGHT, AND
C      MASS FRACTION
       WRITE (6,4)
     4 FORMAT (4HOU4 ,
      114HBURN TIME ONE ,2X,14HBURN TIME FOUR,2X,14HWP ONE        ,2X,
      114HWP FOUR       ,2X,14HHARD ONE      ,2X,14HHARD FOUR     ,2X,
      114HM F ONE       ,2X,14HM F FOUR      /4H U5 ,
      114HBURN TIME TWO ,2X,14HBURN TIME FIVE,2X,14HWP TWO        ,2X,
      114HWP FIVE       ,2X,14HHARD TWO      ,2X,14HHARD FIVE     ,2X,
      114HM F TWO       ,2X,14HM F FIVE      /4H U6 ,
      114HBURN TIME THRE,2X,14HBURN TIME SIX ,2X,14HWP THREE      ,2X,
      114HWP SIX        ,2X,14HHARD THREE    ,2X,14HHARD SIX      ,2X,
      114HM F THREE     ,2X,14HM F SIX       )
       RETURN
C
C      IF N EQUAL 2, WRITE DATA CORRESPONDING TO FIRST HEADING
     5 GO TO (7,6),N
     6 WRITE (6,10 ) THRUST(1),THRUST(4),FLOMX(1),FLOMX(4),HARD(1),HARD(4
      1),PROP(1),PROP(4),THRUST(2),THRUST(5),FLOMX(2),FLOMX(5),HARD(2),
      1HARD(5),PROP(2),PROP(5),THRUST(3),THRUST(6),FLOMX(3),FLOMX(6),
      1HARD(3),HARD(6),PROP(3),PROP(6)
       RETURN
C
C      CALL ORBEL TO CALCULATE ORBIT ELEMENTS
     7 CALL ORBEL
C
C      CALCULATE DATA TO BE PRINTED FROM RAW DATA
C
C      EXPRESS THRUST ANGLE IN DEGREES
       SID = PSI * CONRTD
C
C      COMPUTE DERIVATIVE OF THRUST ANGLE
       PSID=(ZLAM2*DZLAM1-ZLAM1*DZLAM2)/(ZLAM1**2+ZLAM2**2)*CONRTD
C
C      COMPUTE PERIGEE ALTITUDE
       PERALT=P/(1.0+E)-RO
C
C      COMPUTE ALTITUDE
       ALT=R-RO
C
       OMEGAD=OMEGA*CONRTD
C
C      COMPUTE FINAL BURNOUT WEIGHT WHEN AN IDEAL DELTAV IS ADDED TO THE
C      LAST PHASE
       ZMASS=RMASS
       IF(NSTAG1.NE.LAST+1) GO TO 8
       ZMASS =RMASS-(1.0+PROP(LAST))*FUELDV-HARD(LAST)-PROP(LAST)*FUEL
      1(LAST)-DROP
C
C
C      CALCULATE THRUST-TO-WEIGHT
     8 TTOW=FDM/G
C
C      CALCULATE INERTIAL TRAVEL ANGLE FOR UPPER PHASE PORTION OF FLIGHT
       FID=PHI*CONRTD
C
C      CALCULATE HORIZONAL VELOCITY
       HORV = OMEGA * R
C
C      CALCULATE INERTIAL FLIGHT PATH ANGLE
```

```
      BETAD=ARCTAN(U,HORV)*CONRTD                                       ZOTP0162
C                                                                       ZOTP0163
      WRITE (6,11 )TIME,ZMASS,SID,ALT,E,BETAD,PERIOD,V,FLOW ,TTOW,      ZOTP0164
     1PSID,PERALT,TP,OMEGAD,FID,U,STEPNO,CAPPA,TKICK,R,TPD,P,THETA,     ZOTP0165
     1HORV                                                              ZOTP0166
C                                                                       ZOTP0167
C     IF NSTAG1 EQUALS NSTAGE, THEN THE TRAJECTORY IS COMPLETE          ZOTP0168
C     AND THE INFORMATION TO BE TABULATED AT THE END OF A               ZOTP0169
C     TRAJECTORY IS COMPUTED AND PRINTED.                               ZOTP0170
C     CALCULATE TOTAL HARDWARE WEIGHT AND MASS FRACTION                 ZOTP0171
      IF(NSTAG1.NE.LAST+1) RETURN                                       ZOTP0172
      LAST1 = LAST - 1                                                  ZOTP0173
      DO 9 J = 1,LAST1                                                  ZOTP0174
C                                                                       ZOTP0175
C     COMPUTE TOTAL HARDWARE WEIGHT FOR ALL BUT LAST PHASE              ZOTP0176
      AMF(J)=0.0                                                        ZOTP0177
      HARD1(J)=HARD(J)+FUEL(J)*PROP(J)                                  ZOTP0178
C                                                                       ZOTP0179
    9 IF(FUEL(J)+HARD1(J).NE.0.0)                                       ZOTP0180
C                                                                       ZOTP0181
C     COMPUTE MASS FRACTION FOR ALL BUT LAST PHASE                      ZOTP0182
     1AMF(J)=FUEL(J)/(FUEL(J)+HARD1(J))                                 ZOTP0183
C                                                                       ZOTP0184
C                                                                       ZOTP0185
C     COMPUTE FUEL FOR LAST PHASE                                       ZOTP0186
      FUEL(LAST)=FUEL(LAST)+FUELDV                                      ZOTP0187
C                                                                       ZOTP0188
C     COMPUTE HARDWARE WEIGHT FOR LAST PHASE                            ZOTP0189
      HARD1(LAST)=HARD(LAST)+PROP(LAST)*FUEL(LAST)                      ZOTP0190
C                                                                       ZOTP0191
C     COMPUTE MASS FRACTION FOR LAST PHASE                              ZOTP0192
      IF(FUEL(LAST).NE.0.0)                                             ZOTP0193
     1AMF(LAST)=FUEL(LAST)/(FUEL(LAST)+HARD1(LAST))                     ZOTP0194
C                                                                       ZOTP0195
C     PRINT OUT SUMMARY OF TRAJECTORY DATA                              ZOTP0196
      WRITE (6,12 )TB(1),TB(4),FUEL(1),FUEL(4),HARD1(1),HARD1(4),AMF(1),ZOTP0197
     1AMF(4),TB(2),TB(5),FUEL(2),FUEL(5),HARD1(2),HARD1(5),AMF(2),AMF(5)ZOTP0198
     1,TB(3),TB(6),FUEL(3),FUEL(6),HARD1(3),HARD1(6),AMF(3),AMF(6)      ZOTP0199
   10 FORMAT(4HOA1 ,7(G14.7,2X),G14.7/                                  ZOTP0200
     14H A2 ,7(G14.7,2X),G14.7/                                         ZOTP0201
     14H A3 ,7(G14.7,2X),G14.7)                                         ZOTP0202
   11 FORMAT(4HOU1 ,7(G14.7,2X),G14.7/                                  ZOTP0203
     14H U2 ,7(G14.7,2X),G14.7/                                         ZOTP0204
     14H U3 ,I5,11X,6(G14.7,2X),G14.7)                                  ZOTP0205
   12 FORMAT(4HOU4 ,7(G14.7,2X),G14.7/                                  ZOTP0206
     14H U5 ,7(G14.7,2X),G14.7/                                         ZOTP0207
     14H U6 ,7(G14.7,2X),G14.7)                                         ZOTP0208
      RETURN                                                            ZOTP0209
      END                                                               ZOTP0210
```

```
      SUBROUTINE OUTPT2(N)
C
C          SUBROUTINE OUTPT2 CONTAINS THE WRITE STATEMENTS WHICH
C     PRINT DATA FROM THE BOOSTER PORTION OF THE PROGRAM.  THE USER
C     HAS HIS CHOICE OF A FIVE LINE PRINTOUT OR A TWO LINE
C     PRINTOUT.
C
      COMMON /RUNG/RUN(125)
      COMMON /CSTAR/ CMA(1000),CMB(1000)
      COMMON /ATABLE/ CME(8000)
      DIMENSION H      (5        ),RB     (5        ),VATM   (5        )
      DIMENSION VX     (5        )
      EQUIVALENCE (ALTB  ,CMB(154)),(ALTE  ,CMA(805)),(AZIMI ,CMA(921))
      EQUIVALENCE (AZIMR ,CMA(922)),(BETAI ,CMA(917)),(BETAR ,CMA(918))
      EQUIVALENCE (CD    ,CMA(812)),(E     ,CMA(905)),(FIXDTK,CMB(071))
      EQUIVALENCE (FLOW  ,CMA(752)),(FM    ,CMA(715)),(G     ,CMA(716))
      EQUIVALENCE (H     ,CMA(769)),(HEAT  ,CMA(401)),(HORV  ,CMB(156))
      EQUIVALENCE (IMODE ,CMB(061)),(MODEC ,CMB(180)),(NKICK ,CME(020))
      EQUIVALENCE (NVAR  ,CMB(073)),(P     ,CMA(906)),(PA    ,CMA(806))
      EQUIVALENCE (PERIOD,CMA(910)),(PHII  ,CMA(911)),(PHIR  ,CMA(912))
      EQUIVALENCE (Q     ,CMA(807)),(QVAL  ,CMA(808)),(RADIUS,CMA(758))
      EQUIVALENCE (RANGE ,CMA(914)),(RB    ,CMA(754)),(RDOT  ,CMB(155))
      EQUIVALENCE (RE    ,CMA(924)),(ROA   ,CMB(062)),(STEPGO,RUN(112))
      EQUIVALENCE (STEPNO,RUN(113)),(THETA ,CMA(909)),(THRUST,CMA(751))
      EQUIVALENCE (TIME  ,CMA(409)),(TP    ,CMA(907)),(TPD   ,CMA(908))
      EQUIVALENCE (TRAVLI,CMA(913)),(VAR   ,CME(501)),(VATM  ,CMA(764))
      EQUIVALENCE (VMACH ,CMA(811)),(VX    ,CMA(759)),(WEIGHT,CMA(402))
      EQUIVALENCE (ZINCLI,CMA(919)),(ZINCLR,CMA(920)),(ZLAT  ,CMA(916))
      EQUIVALENCE (ZLONG ,CMA(915)),(ZNODFI,CMA(923))
      INTEGER STEPGO,STEPNO,FIXDTK
      DATA PALPHA/0.0/
      DATA   RADDEG/57.2957795/
      IF(N.NE.0) GO TO 6
      GO TO (9,1,3),MODEC
C
C     THE SHORT PRINTOUT CONTAINS THE FOLLOWING VARIABLES.
C     TIME              TIME FROM LAUNCH                        SEC
C     WEIGHT            INSTANTANEOUS WEIGHT                    LB
C     RAD VELOCITY      RADIAL VELOCITY                         FT/SEC
C     HOR VELOCITY      HORIZONAL VELOCITY
C     ALTITUDE          ALTITUDE ABOVE OBLATE EARTH             FT
C     INR. VELOCITY     INERTIAL VELOCITY                       FT/SEC
C     INR. GAMMA        INERTIAL FLIGHT PATH ANGLE              DEG
C     FLOW              FLOW RATE                               LB/SEC
C     STEPGO            NUMBER OF GOOD INTEGRATION STEPS
C                       UP TO AND INCLUDING PRINTOUT
C     STEPNO            NUMBER OF BAD INTEGRATION STEPS
C                       UP TO PRINTOUT
C     RADIUS            DISTANCE TO CENTER OF EARTH             FT
C     DRAG              AERODYNAMIC FORCE ALONG THE VEHICLE     LB
C                       LONGITUDINAL AXIS
C     PRESSURE          ATMOSPHERIC PRESSURE                    LB/FT**2
C     HEAT INTEGRAL     TIME INTEGRAL FROM LIFTOFF OF THE       LB/FT
C                       PRODUCT OF DYNAMIC PRESSURE AND
C                       VELOCITY RELATIVE TO AN EARTH FIXED
C                       COORDINATE SYSTEM.
```

```
C       MACH NUMBER       RELATIVE VELOCITY DIVIDED BY THE                            ZOTP0057
C                         SPEED OF SOUND                                              ZOTP0058
C       Q                 DYNAMIC PRESSURE                          LB/FT**2          ZOTP0059
C       THRUST            THRUST                                    LB                ZOTP0060
      1 WRITE (6,2)                                                                   ZOTP0061
      2 FORMAT ( 4HOB1 ,                                                              ZOTP0062
       114HTIME          ,2X,14HWEIGHT          ,2X,14HRAD. VELOCITY ,2X,             ZOTP0063
       214HHOR. VELOCITY ,2X,14HALTITUDE        ,2X,14HINR. VELOCITY ,2X,             ZOTP0064
       314HINR. GAMMA    ,2X,14HFLOW            /4H B2 ,                              ZOTP0065
       414HSTEPGO+STEPNO ,2X,14HRADIUS          ,2X,14HDRAG          ,2X,             ZOTP0066
       514HPRESSURE      ,2X,14HHEAT INTEGRAL  ,2X,14HMACH NUMBER    ,2X,             ZOTP0067
       614HQ             ,2X,14HTHRUST          )                                     ZOTP0068
C                                                                                     ZOTP0069
        GO TO 9                                                                       ZOTP0070
C                                                                                     ZOTP0071
C       THE LONG PRINTOUT CONTAINS THE FOLLOWING VARIABLES                            ZOTP0072
C       TIME              TIME FROM LAUNCH                          SEC               ZOTP0073
C       LATITUDE          GEOCENTRIC LATITUDE,POSITIVE NORTH OF DEG                   ZOTP0074
C                         EQUATOR                                                     ZOTP0075
C       RADIUS            DISTANCE TO THE GEOCENTRIC CENTER OF      FT                ZOTP0076
C                         THE EARTH                                                   ZOTP0077
C       INR. VELOCITY     INERTIAL VELOCITY                         FT/SEC            ZOTP0078
C       PERIOD            PERIOD OF ELLIPTICAL ORBIT                SEC               ZOTP0079
C       SEM LAT REC       SEMILATUS RECTUM                          FT                ZOTP0080
C       WEIGHT            INSTANTANEOUS WEIGHT OF CONFIGURATION LB                    ZOTP0081
C       ACCELLERATION     AXIAL ACCELLERATION, THRUST LESS          G S               ZOTP0082
C                         DRAG DIVIDED BY WEIGHT                                      ZOTP0083
C       STEPGO            NUMBER OF GOOD INTEGRATION STEPS                            ZOTP0084
C                         UP TO AND INCLUDING PRINTOUT                                ZOTP0085
C       STEPNO            NUMBER OF BAD INTEGRATION STEPS                             ZOTP0086
C                         UP TO PRINTOUT                                              ZOTP0087
C       LONGITUDE         LONGITUDE, POSITIVE EAST OF               DEG               ZOTP0088
C                         GREENWICH                                                   ZOTP0089
C       ALTITUDE          ALTITUDE ABOVE THE OBLATE EARTH           FT                ZOTP0090
C       INR GAMMA         INERTIAL FLIGHT PATH ANGLE                DEG               ZOTP0091
C       TRUE ANOMALY      TRUE ANOMALY                              DEG               ZOTP0092
C       ECCENTRICITY      ECCENTRICITY                                                ZOTP0093
C       FLOW              FLOW RATE                                 LB/SEC            ZOTP0094
C       THRUST            THRUST                                    LB                ZOTP0095
C       PSI               ANGLE BETWEEN THRUST VECTOR AND LOCAL DEG                   ZOTP0096
C                         HORIZONTAL, MEASURED POSITIVE FOR                           ZOTP0097
C                         THRUST ORIENTED ABOVE THE HORIZONTAL                        ZOTP0098
C       INR. AZIMUTH      AZIMUTH OF INERTIAL VELOCITY VECTOR,      DEG               ZOTP0099
C                         THE ANGLE BETWEEN PROJECTION OF THE                         ZOTP0100
C                         INERTIAL VELOCITY VECTOR INTO THE                           ZOTP0101
C                         AZIMUTH PLANE (PLANE PERPENDICULAR TO                       ZOTP0102
C                         RADIUS VECTOR) AND NORTH DIRECTION,                         ZOTP0103
C                         POSITIVE CLOCKWISE FROM NORTH                               ZOTP0104
C       RANGE             SURFACE RANGE FROM LAUNCH PAD             NM                ZOTP0105
C       REL. VELOCITY     RELATIVE VELOCITY, REFERENCED TO A        FT/SEC            ZOTP0106
C                         COORDINATE SYSTEM FIXED WITH                                ZOTP0107
C                         THE ROTATING EARTH.                                         ZOTP0108
C       NODE              THE LONGITUDE OF THE POINT AT WHICH       DEG               ZOTP0109
C                         PLANE OF THE ORBIT CROSSES THE                              ZOTP0110
C                         EQUATORIAL PLANE FROM SOUTH TO NORTH                        ZOTP0111
C       INR INCLINATION   INERTIAL INCLINATION, THE ANGLE           DEG               ZOTP0112
C                         MEASURED FROM THE EQUATORIAL PLANE                          ZOTP0113
C                         TO THE ORBIT PLANE, COUNTERCLOCK-                           ZOTP0114
C                         WISE AT THE ASCENDING NODE.                                 ZOTP0115
C       RAD. VEL          RADIAL VELOCITY                           FT/SEC            ZOTP0116
```

```
C      INR. TRVL ANGL   THE ANGLE BETWEEN THE LAUNCH RADIUS   DEG              ZOTP0117
C                       VECTOR AND THE POSITION VECTOR                         ZOTP0118
C      PSID             TIME RATE OF CHANGE OF PSI            DEG/SEC          ZOTP0119
C      REL. AXIMUTH     AZIMUTH OF RELATIVE VELOCITY VECTOR,  DEG              ZOTP0120
C                       THE ANGLE BETWEEN PROJECTION OF THE                    ZOTP0121
C                       RELATIVE VELOCITY VECTOR INTO THE                      ZOTP0122
C                       AZIMUTH PLANE (PLANE PERPENDICULAR TO                  ZOTP0123
C                       RADIUS VECTOR) AND NORTH DIRECTION,                    ZOTP0124
C                       POSITIVE CLOCKWISE FROM NORTH                          ZOTP0125
C      ALPHA            THE ANGLE OF ATTACK IN THE PITCH      DEG              ZOTP0126
C                       PLANE                                                  ZOTP0127
C      REL GAMMA        RELATIVE FLIGHT PATH ANGLE            DEG              ZOTP0128
C      TIME PER DEP     TIME OF PERIGEE DEPARTURE             SEC              ZOTP0129
C      REL. INCLIN.     RELATIVE INCLINATION, ANGLE MEASURED  DEG              ZOTP0130
C                       FROM THE EQUATORIAL PLANE TO THE                       ZOTP0131
C                       ORBIT PLANE, COUNTER CLOCKWISE AT THE                  ZOTP0132
C                       ASCENDING NODE                                         ZOTP0133
C      HOR. VELOCITY    HORIZONTAL VELOCITY                   FT/SEC           ZOTP0134
C      REL TRVL ANGL    THE CENTRAL ANGLE BETWEEN THE VEHICLE DEG              ZOTP0135
C                       AND THE LAUNCH SITE                                    ZOTP0136
C      DRAG             AERODYNAMIC FORCE ALONG THE VEHICLE   IB               ZOTP0137
C                       LONGITUDINAL AXIS                                      ZOTP0138
C      PRESSURE         ATMOSPHERIC PRESSURE                  LB/FT**2         ZOTP0139
C      HEAT INTEGR.     TIME INTEGRAL FROM LIFTOFF OF THE     LB/FT            ZOTP0140
C                       PRODUCT OF DYNAMIC PRESSURE AND                        ZOTP0141
C                       VELOCITY RELATIVE TO AN EARTH FIXED                    ZOTP0142
C                       COORDINATE SYSTEM.                                     ZOTP0143
C      MACH NUMBER      MACH NUMBER, RELATIVE VELOCITY DIVIDED                 ZOTP0144
C                       BY THE SPEED OF SOUND                                  ZOTP0145
C      DRAG COEFF       DRAG COEFFICIENT                                       ZOTP0146
C      DYNAMIC PRES     DYNAMIC PRESSURE                      LB/FT**2         ZOTP0147
C      TIME OF PERIGE   TIME OF PERIGEE                       SEC              ZOTP0148
C      INR. TRAVEL      SURFACE DISTANCE TRAVERSED OVER A     NM               ZOTP0149
C                       NON-ROTATING EARTH                                     ZOTP0150
    3 WRITE (6,4)                                                              ZOTP0151
      WRITE (6,5)                                                              ZOTP0152
    4 FORMAT ( 4HOB1 ,                                                         ZOTP0153
     114HTIME           ,2X,14HLATITUDE      ,2X,14HRADIUS         ,2X,        ZOTP0154
     214HINR. VELOCITY  ,2X,14HPERIOD        ,2X,14HSEM LAT REC    ,2X,        ZOTP0155
     314HWEIGHT         ,2X,14HACCELLERATION /4H B2 ,                          ZOTP0156
     414HSTEPGO+STEPNO  ,2X,14HLONGITUDE     ,2X,14HALTITUDE       ,2X,        ZOTP0157
     514HINR. GAMMA     ,2X,14HTRUE ANOMALY  ,2X,14HECCENTRICITY   ,2X,        ZOTP0158
     614HFLOW           ,2X,14HTHRUST        /4H B3 ,                          ZOTP0159
     714HPSI            ,2X,14HINR. AZIMUTH  ,2X,14HRANGE          ,2X,        ZOTP0160
     814HREL. VELOCITY  ,2X,14HNODE          ,2X,14HINR. INCLIN.   ,2X,        ZOTP0161
     914HRAD. VELOCITY  ,2X,14HINR TRVL ANGL )                                 ZOTP0162
    5 FORMAT (4H B4 ,                                                          ZOTP0163
     114HPSID           ,2X,14HREL. AZIMUTH  ,2X,14HALPHA          ,2X,        ZOTP0164
     214HREL. GAMMA     ,2X,14HTIME PER DEP  ,2X,14HREL. INCLIN.   ,2X,        ZOTP0165
     314HHOR VELOCITY   ,2X,14HREL TRVL ANGL /4H B5 ,                          ZOTP0166
     414HDRAG           ,2X,14HPRESSURE      ,2X,14HHEAT INTEGR.   ,2X,        ZOTP0167
     514HMACH    NO     ,2X,14HDRAG COEFF    ,2X,14HDYNAM. PRES.   ,2X,        ZOTP0168
     614HTIME OF PERIGE,2X,14HINR.TRAVEL       )                               ZOTP0169
C                                                                              ZOTP0170
      RETURN                                                                   ZOTP0171
C                                                                              ZOTP0172
C     COMPUTE RADIAL VELOCITY                                                  ZOTP0173
    6 RDOT = DOT(RB,VX)/RB(5)                                                  ZOTP0174
C                                                                              ZOTP0175
C     COMPUTE ANGULAR MOMENTUM VECTOR                                          ZOTP0176
```

```
      CALL CONVT (RB,VX,H)                                              ZOTP0177
C                                                                       ZOTP0178
C     COMPUTE INERTIAL FLIGHT PATH ANGLE                                ZOTP0179
      BETAI = ARCTAN(RDOT,H(5)/RB(5))*RADDEG                            ZOTP0180
C                                                                       ZOTP0181
C     COMPUTE DRAG                                                      ZOTP0182
      DRAG=CD*QVAL*WEIGHT/G                                             ZOTP0183
C                                                                       ZOTP0184
C     COMPUTE HORIZONTAL VELOCITY                                       ZOTP0185
      HORV = H(5)/RB(5)                                                 ZOTP0186
C                                                                       ZOTP0187
C                                                                       ZOTP0188
C     COMPUTE ALTITUDE TO BE STORED                                     ZOTP0189
      ALTB=RADIUS -ROA                                                  ZOTP0190
C                                                                       ZOTP0191
C     CALL LOAD TO STORE DATA IF DESIRED                                ZOTP0192
      IF(FIXDTK.EQ.3.AND.IMODE.EQ.1) CALL LOAD(VAR,NVAR,NKICK)          ZOTP0193
C                                                                       ZOTP0194
C     CHOOSE OUTPUT FORM DESIRED, SEE ABOVE                             ZOTP0195
      GO TO (9,8,7),MODEC                                               ZOTP0196
C     FIVE LINE OUTPUT                                                  ZOTP0197
C     CALL ORBEL TO ACQUIRE ORBIT ELEMENT DATA                          ZOTP0198
    7 CALL ORBEL2                                                       ZOTP0199
C                                                                       ZOTP0200
C     COMPUTE THRUST-TO-WEIGHT                                          ZOTP0201
      TTW=(THRUST-DRAG)/WEIGHT                                          ZOTP0202
C     COMPUTE PERIGEE ALTITUDE                                          ZOTP0203
      PERALT = P/(1.0+E)-RE                                             ZOTP0204
C                                                                       ZOTP0205
C     COMPUTE PSID                                                      ZOTP0206
      PSID = H(5)/RADIUS/RADIUS*(1.0-FM/RADIUS/VX(4))*RADDEG            ZOTP0207
C                                                                       ZOTP0208
C     SINCE ANGLE-OF-ATTACK EQUALS ZERO, PSI = RELATIVE FLIGHT PATH     ZOTP0209
C     ANGLE                                                             ZOTP0210
      QSI=BETAR                                                         ZOTP0211
C                                                                       ZOTP0212
      WRITE(6,10) TIME,ZLAT,RADIUS,VX(5),PERIOD,P,WEIGHT,TTW,STEPGO,    ZOTP0213
     1STEPNO,ZLONG,ALTE,BETAI,THETA,E,FLOW,THRUST,QSI,AZIMI,RANGF,VATM(5ZOTP0214
     2),ZNODFI,ZINCLI,RDOT,PHII,PSID,AZIMR,PALPHA,BETAR,TPD,ZINCLR,HORV,ZOTP0215
     3PHIR,DRAG,PA,HEAT,VMACH ,CD,Q,TP,TRAVLI                           ZOTP0216
      GO TO 9                                                           ZOTP0217
    8 WRITE (6,11) TIME,WEIGHT,RDOT,HORV,ALTE,VX(5),BETAI,FLOW,STEPGO,  ZOTP0218
     1STEPNO,RADIUS,DRAG,PA,HEAT,VMACH ,Q,THRUST                        ZOTP0219
    9 RETURN                                                            ZOTP0220
   10 FORMAT ( 4HOB1 ,                   7(G14.7,2X),G14.7/             ZOTP0221
     14H B2 ,I5,4X,I5,2X,6(G14.7,2X),G14.7/                             ZOTP0222
     24H B3 ,7(G14.7,2X),G14.7/                                         ZOTP0223
     34H B4 ,7(G14.7,2X),G14.7/                                         ZOTP0224
     44H B5 ,7(G14.7,2X),G14.7)                                         ZOTP0225
   11 FORMAT ( 4HOB1 ,                   7(G14.7,2X),G14.7/             ZOTP0226
     14H B2 ,I5,4X,I5,2X,6(G14.7,2X),G14.7)                             ZOTP0227
      END                                                               ZOTP0228
```

```
C
C         THIS ROUTINE ZEROS THE DATA WHICH MUST BE ZEROED BEFORE
C      PROGRAM EXECUTION.
C
      BLOCK DATA
C     PERDAT
      DIMENSION COMPA (3      ),DRAG   (3      ),EXITS (6      )
      DIMENSION FLOMX (6      ),FORCE  (3      ),FORCES(6      )
      COMMON/CSTAR/CMA(1000),CMB(1000)
      DIMENSION FYD   (5      ),HARD   (6      ),HARDB (6      )
      DIMENSION NOPT  (6      ),NOUT   (6   ,3 ),NSETS (20     )
      DIMENSION OBLAT (3      ),PROP   (6      ),TB    (6      )
      DIMENSION TBOOST(6      ),THRUST(6       ),WPMAX (6      )
      DIMENSION WTFLOW(6      ),XINPT (100    )
      EQUIVALENCE (BETA  ,CMA(880)),(CAPPA ,CMA(891)),(CD    ,CMA(812))
      EQUIVALENCE (COMPA ,CMA(783)),(DELTAV,CMA(861)),(DRAG  ,CMA(777))
      EQUIVALENCE (DROP  ,CMA(863)),(ENERGY,CMA(892)),(EXITS ,CMA(727))
      EQUIVALENCE (FLOMX ,CMA(837)),(FORCE ,CMA(774)),(FORCES,CMA(739))
      EQUIVALENCE (FYD   ,CMB(036)),(HARD  ,CMA(843)),(HARDB ,CMA(721))
      EQUIVALENCE (ITERPD,CMB(060)),(MASH  ,CMB(064)),(NOPT  ,CMA(819))
      EQUIVALENCE (NOUT  ,CMB(183)),(NSETS ,CMB(221)),(OBLAT ,CMA(780))
      EQUIVALENCE (PA    ,CMA(806)),(PROP  ,CMA(849)),(QVAL  ,CMA(808))
      EQUIVALENCE (RESERV,CMA(862)),(TB    ,CMA(825)),(TBOOST,CMA(745))
      EQUIVALENCE (THRUST,CMA(831)),(VMACH ,CMA(811)),(WPMAX ,CMA(855))
      EQUIVALENCE (WTFLOW,CMA(733)),(XINPT ,CMA(601))
      DATA
     1BETA/0.0/,
     1CAPPA/0.0/,
     1CD/0.0/,
     1(COMPA(J),J=1,3)/3*0.0/,
     1DELTAV/0.0/,
     1(DRAG (J),J=1,3)/3*0.0/,
     1DROP/0.0/,
     1ENERGY/0.0/,
     1(EXITS (J),J=1,5)/5*0.0/,
     1(FLOMX (J),J=1,6)/6*0.0/,
     1(FORCE(J),J=1,3)/3*0.0/,
     1(FORCES(J),J=1,5)/5*0.0/,
     1(FYD(I),I=1,5)/5*0.0/,
     1(HARD  (J),J=1,6)/6*0.0/,
     1(HARDB (J),J=1,5)/5*0.0/
      DATA
     1ITERPD/0/,
     1MASH/0/,
     1(NOPT(J),J=1,6)/6*0/,
     1((NOUT(I,J),I=1,6),J=1,3)/18*0/,
     1NSETS/20*0/,
     1(OBLAT(J),J=1,3)/3*0.0/,
     1PA/0.0/,
     1(PROP  (J),J=1,6)/6*0.0/
      DATA
     1QVAL/0.0/,
     1RESERV/0.0/,
     1(TB(J),J=1,6)/6*0.0/,
     1(TBOOST(J),J=1,5)/5*0.0/,
     1(THRUST(J),J=1,6)/6*0.0/,
     1VMACH/0.0/,
     1(WPMAX(J),J=1,6)/6*0.0/,
     1(WTFLOW(J),J=1,5)/5*0.0/,
     1(XINPT(I),I=1,10)/10*0.0/
      END
```

```
      SUBROUTINE PERTB (DX,ICHECK)
C
C          SUBROUTINE PERTB COMPUTES THE PERTURBATION SIZE REQUIRED
C     TO OBTAIN MEANINGFUL FINITE DIFFERENCES FOR USE IN THE
C     ITERATION SCHEME.
C          IF A SET OF PERTURBATIONS HAS BEEN RUN, THE PERTURBATION
C     SIZE FOR THE NEXT SET OF PERTURBATIONS IS ADJUSTED SUCH THAT
C     THE DIFFERENCE INDICATOR (COMPUTED IN THE SAME MANNER AS THE
C     ERROR INDICATOR IN MAIN) IS NOMINALLY EQUAL TO 1.0E-04 AND
C     IS ACCEPTABLE IF IT IS BETWEEN 5.0E-04 AND 5.0E-05.
C          IF NO SET OF PERTURBATIONS IS AVAILABLE, (CHANGE IN THE
C     RADIUS)/RADIUS IS TREATED IN THE SAME MANNER AS THE DIFFERENCE
C     INDICATOR AS DESCRIBED ABOVE.
C     DX                PERTURBATION FACTOR (SEE MAIN)
C     ICHECK            FLAG INDICATING ACCEPTABILITY OF PERTURBATION.
C
      COMMON /CSTAR/ CMA(1000),CMB(1000)
      COMMON /PERB/NA,N,SF(5),COUNT,RSTO
      DIMENSION FY     (6   ,6   ),XINPT (100     )
      EQUIVALENCE (FY     ,CMA(943)),(R      ,CMA(402)),(XINPT ,CMA(601))
      INTEGER COUNT
C
C     IF COUNT EQUALS ZERO, NO PERTURBATION SET IS AVAILABLE
      IF(COUNT.EQ.0) GO TO 2
      DIFRAD=0.0
C
C     COMPUTE DIFFERENCE INDICATOR AS IN MAIN
      DO 1  J = 1,NA
    1 DIFRAD=DIFRAD+((FY(N,J)-FY(1,J))*SF(J))**2
      DIFRAD=SQRT(DIFRAD)/XINPT(1)
C
      GO TO 3
C     COMPUTE INDICATOR BASED ON RADIUS CHANGE
    2 DIFRAD=ABS((R-RSTO)/RSTO)
    3 IF(ICHECK.NE.0) GO TO 4
C
C     LINEAR INTERPOLATION TO ADJUST PERTURBATION FACTOR
      DX=DX/DIFRAD*1.0E-04
C
C     CHECK ACCEPTABILITY OF PERTURBATION SIZE
    4 IF((DIFRAD.LT.5.0E-4.AND.DIFRAD.GT.5.0E-5).OR.ICHECK.EQ.4) GO TO 6
C
      IF(ICHECK.EQ.0) GO TO 5
C
C     QUADRATIC INTERPOLATION TO ADJUST PERTURBATION FACTOR
      DX=((DX*DIFSTO-DIFRAD*DXSTO)*1.0E-04+(DIFRAD**2*DXSTO-DIFSTO**2
     1*DX))*1.0E-04/(DIFRAD*DIFSTO*(DIFRAD-DIFSTO))
C
C     STORE INFORMATION FOR QUADRATIC INTERPOLATION
    5 DXSTO=DX
      DIFSTO=DIFRAD
      ICHECK=ICHECK+1
C
      RETURN
C
C     IF PERTURBATION SIZE IS ACCEPTABLE, SET ICHECK TO ZERO
C     AND RETURN.
    6 ICHECK=0
C
      RETURN
      END
```

```
      FUNCTION QUAD  (X,IC)                                             ZQUA0001
C                                                                       ZQUA0002
C           FUNCTION QUAD COMPUTES ANY VARIABLE, QUAD, AS A QUADRATIC   ZQUA0003
C     FUNCTION OF X.  QUAD = A + BX +CXX.  THERE MAY BE SEVERAL         ZQUA0004
C     SETS OF COEFFICIENTS, EACH SET BELONGING TO A PARTICULAR          ZQUA0005
C     REGION OF X.  THE COEFN ARRAY IS ARRANGED AS-                     ZQUA0006
C             X1= A1, B1, C1, X2, A2, B2, C2, X3, A3, B3, C3, X4,...    ZQUA0007
C     WHERE A1, B1, C1 ARE THE COFFFICIENTS TO BE USED FOR X            ZQUA0008
C     BETWEEN X1 AND X2, ETC. AND X1 IS LESS THAN X2, X2 IS LESS        ZQUA0009
C     THAN X3, X3 IS LESS THAN X4, ETC.                                 ZQUA0010
C           IC IDENTIFIES WHICH DEPENDENT VARIABLE IS BEING             ZQUA0011
C     SOUGHT.  ICC(IC) DEFINE THE STARTING LOCATIONS IN THE COEFN       ZQUA0012
C     ARRAY FOR VARIABLE X.                                             ZQUA0013
C                                                                       ZQUA0014
      COMMON /CSTAR/ CMA(1000),CMB(1000)                                ZQUA0015
      DIMENSION COEFN (500     ),ICC    (20       )                     ZQUA0016
      EQUIVALENCE (COEFN ,CMB(501)),(ICC    ,CMB(201))                  ZQUA0017
C                                                                       ZQUA0018
      I=ICC(IC)                                                         ZQUA0019
    1 IF(X-COEFN(I)) 2,3,3                                              ZQUA0020
    2 I = I-4                                                           ZQUA0021
      GO TO 1                                                           ZQUA0022
    3 IF(X-COEFN(I+4)) 5,5,4                                            ZQUA0023
    4 I = I+4                                                           ZQUA0024
      GO TO 3                                                           ZQUA0025
    5 QUAD = COEFN(I+1)+X*(COEFN(I+2)+X*COEFN(I+3))                     ZQUA0026
      ICC(IC)=I                                                         ZQUA0027
      RETURN                                                            ZQUA0028
      END                                                               ZQUA0029
```

```
      SUBROUTINE RENDER (N)
C
C          SUBROUTINE RENDER CONTAINS THE CALLING SEQUENCES FOR
C     THE ROUTINES WHICH READ AND PUNCH BINARY DATA.
C          IF N IS EQUAL TO ONE, BINARY DATA IS PRESENT AND MUST BE
C     READ.  IF N IS TWO, A NEW BOOSTER TABLE IS TO BE GENERATED.
C
      COMMON/ATABLE/CME(8000)
      COMMON /CSTAR/ CMA(1000),CMB(1000)
      DIMENSION FIRST (28      ),FOURTH(7500    ),SECOND(2       )
      DIMENSION THIRD (300     )
      EQUIVALENCE (FIRST ,CME(001)),(FOURTH,CME(501)),(IKCKST,CMB(066))
      EQUIVALENCE (IKICK ,CME(200)),(NVAR  ,CMB(073)),(SECOND,CME(199))
      EQUIVALENCE (THIRD ,CME(201))
      IF(N.EQ.2) GO TO 2
C
C     BINARY READ PORTION
C
C     READ VERTICAL KICK DATA
      READ (5,1) FIRST(1)
    1 FORMAT (A6)
      CALL BCREAD (FIRST(1),FIRST(28))
C
C     READ DELTK (KICK ANGLE SPACING) AND IKICK (NUMBER OF KICK ANGLES)
      CALL BCREAD (SECOND(1),SECOND(2))
C
C     READ LIST OF KICK ANGLES IN TABLE
      IKCKST = IKICK
      K = 3*IKICK
      L = 25*NVAR*IKICK
      CALL BCREAD (THIRD(1),THIRD(K))
C
C     READ BOOSTER BURNOUT CONDITIONS
      CALL BCREAD (FOURTH(1),FOURTH(L))
C
      RETURN
C     BINARY WRITE PORTION
C
C     IF IKICK EQUALS IKCKST, THEN NO NEW KICK ANGLES HAVE BEEN
C     RUN TO ADD TO THE TABLE.
    2 IF (IKICK .EQ. IKCKST) RETURN
C
C     IF IKCKST IS EQUAL TO ZERO, A NEW BOOSTER HAS BEEN RUN
C     AND THE VERTICAL RISE DATA MUST BE PUNCHED.
      IF(IKCKST.NE.0) GO TO 4
      WRITE (6,3)
C
      CALL BCDUMP (FIRST(1),FIRST(28))
C
    3 FORMAT(32H$DUMMY CARD INDICATING NEW TABLE,40X,8H   DUMMY)
    4 IKCKST = IKICK
C
C     PUNCH DELTK AND IKICK
      CALL BCDUMP (SECOND(1),SECOND(2))
C
C     PUNCH LIST OF KICK ANGLES
```

```
      K = 3*IKICK                                                     ZRND0057
      L = 25*NVAR*IKICK                                               ZRND0058
      CALL BCDUMP (THIRD(1),THIRD(K))                                 ZRND0059
C                                                                     ZRND0060
C     PUNCH BOOSTER BURNOUT CONDITIONS                                ZRND0061
      CALL BCDUMP (FOURTH(1),FOURTH(L))                               ZRND0062
C                                                                     ZRND0063
      RETURN                                                          ZRND0064
      END                                                             ZRND0065
```

```
      SUBROUTINE RUNGEK (EQUATE,OUTPUT)
C
C          SUBROUTINE RUNGEK CONTROLS THE INTEGRATION SCHEME.
C
      COMMON /RUNG/RUN(125)
      COMMON /CSTAR/ CMA(1000),CMB(1000)
      DIMENSION AK     (3       ),AW     (4        ),OLDINC(100      )
      DIMENSION X      (100     ),XDOT   (100      ),XDOTPM(100,2    )
      DIMENSION XINC   (100     ),XK     (100      ),XPRIM (100,2    )
      EQUIVALENCE (A1    ,RUN(101)),(A2    ,RUN(102)),(DELMAX,CMA(702))
      EQUIVALENCE (DELSTO,RUN(104)),(DELT  ,CMA(701)),(DEL1  ,RUN(103))
      EQUIVALENCE (ERLIMT,CMA(706)),(ERLOG ,CMA(707)),(F2    ,RUN(105))
      EQUIVALENCE (H2    ,RUN(106)),(I     ,RUN(107)),(LAST  ,CMA(711))
      EQUIVALENCE (MODOUT,CMA(714)),(MODS  ,CMA(712)),(NEQ   ,CMA(709))
      EQUIVALENCE (NST   ,CMA(708)),(NSTAGE,CMA(710)),(NSTAG1,RUN(115))
      EQUIVALENCE (NSTEP1,RUN(108)),(NSTEP2,RUN(109)),(NSTEP3,RUN(110))
      EQUIVALENCE (RATIO ,RUN(111)),(SCRIBE,RUN(114)),(STEPGO,RUN(112))
      EQUIVALENCE (STEPMX,CMA(705)),(STEPNO,RUN(113)),(STEPS ,CMA(704))
      EQUIVALENCE (TMIN  ,CMA(703)),(X     ,CMA(401)),(XDOT  ,CMA(501))
      EQUIVALENCE (XINC  ,RUN(001)),(XPRIM ,CMA(001))
      DOUBLE PRECISION XPRIM
      INTEGER STEPGO,STEPNO,SCRIBE,STEPS,STEPMX,FAIL
      DATA FAIL/0/
      DATA(AK(I),I=1,3)/0.5,0.5,1.0/,(AW(I),I=1,4)/.166666666,.333333333
     1,.333333333,.166666666/
      EXTERNAL OUTPUT
C
C
C
C     INITIALIZATIONS
      NSTAG1=NSTAGE
      STEPGO = 0
      STEPNO=0
      SCRIBE = 1
      NSTEP1=1
      NSTEP2=2
      NSTEP3=2
      DELSTO=0.0
      DEL1=DELMAX
C
C     PART 1. SET UP THE STARTING SEQUENCE FOR ERROR CONTROL AND DELAY
C     CHECKING THE ERROR UNTIL TWO STEPS ARE COMPLETED.  THE ASSIGNED GO
C     TOS NSTART AND IBEGIN CONTROL STARTING.
    1 I=1
    2 H2 = DELT
    3 DO 4J=NST,NEQ
      XPRIM(J,2)=XPRIM(J,1)
    4 X(J)=SNGL(XPRIM(J,1))
      NSTART = 0
      DELT = DELT/2.
      CALL EQUATE
      CALL COAST
      MODOUT=MODOUT
      GO TO (5,5,5,6,6,7),MODOUT
    5 NSTEP2=1
    6 IF(STEPGO*SCRIBE.NE.0  ) GO TO 7
```

```
      CALL OUTPUT(1)
      SCRIBE = 1
    7 ASSIGN 18 TO NSTART
      DO 8  J=NST,NEQ
      XDOTPM(J,1) = XDOT(J)
      XINC(J) = 0.
    8 CONTINUE
    9 KSUB = 1
      ASSIGN 13 TO N
C
C     PART 2.  RUNGE-KUTTA SUBINTERVAL SCHEME.  EQUATE PRODUCES THE
C     NECESSARY DERIVATIVES XDOT(J).
   10 DO 11 J = NST,NEQ
      XK(J) = XDOT(J) * DELT
      XINC(J) = XINC(J) + AW(KSUB)*XK(J)
   11 X(J) = SNGL(XPRIM(J,2))+AK(KSUB)*XK(J)
   12 CALL EQUATE
      GO TO N,(13,14,15,17)
C
C     PART 3. SUBINTERVALS 2, 3, AND 4, TO STATEMENT 16 FINISH A
C     RUNGE-KUTTA STEP AND INCREMENT XPRIM(J,2) IN DOUBLE PRECISION.
   13 KSUB = 2
      ASSIGN 14 TO N
      GO TO 10
   14 KSUB = 3
      ASSIGN 15 TO N
      GO TO 10
   15 DO 16 J= NST,NEQ
      XINC(J) = XINC(J) + AW(4) *XDOT(J) * DELT
      XPRIM(J,2)=XPRIM(J,2)+DBLF(XINC(J))
      X(J) = SNGL(XPRIM(J,2))
   16 CONTINUE
C
C     PART 4. BEGIN A NEW RUNGA-KUTTA STEP.  THIS ALSO GIVES DERIVATIVES
C     FOR THE LOWER ORDER INTEGRATION CHECK.
      ASSIGN 17 TO N
      GO TO 12
   17 GO TO NSTART,(24,20,18)
C
C     PART 5.  STARTING PHASE PROGRAM.
C     PART 5A. THIS SECTION COMPLETES THE FIRST STEP OF STARTING PHASE.
   18 ASSIGN 20 TO NSTART
      DO 19 J = NST,NEQ
      OLDINC(J)=XINC(J)
      XINC(J)=0.
      XDOTPM(J,2) = XDOT(J)
   19 CONTINUE
      GO TO 9
C
C     PART 5B. MAX ERROR TEST--STARTING ONLY--CHECK THE MAX ERROR AND
C     EITHER ENTER RUNNING MODE OR REPEAT START WITH SMALLER STEP.
   20 GO TO (21,29,29,29),I
   21 DO 22 J = NST,NEQ
   22 XINC(J) =(XINC(J)+OLDINC(J))*3.-(XDOTPM(J,1)+XDOTPM(J,2)*4.
     1+XDOT(J))*DELT
      CALL ERRORZ
      IF(E2-ERLIMT) 23,23,33
   23 ASSIGN 24 TO NSTART
      ASSIGN 9  TO IBEGIN
      A1 = A2
```

```
      GO TO 29                                                          ZRUN0117
C                                                                       ZRUN0118
C     PART 6.  RUNNING PHASE PROGRAM.                                   ZRUN0119
C     PART 6A. CHECK THE INTEGRATION BY INTEGRATING OVER THE LAST       ZRUN0120
C     RUNGE KUTTA STEP BUT USE DOTS FOR LAST TWO INTERVALS, OLDDEL      ZRUN0121
C     AND DELT RESPECTIVELY.  STATEMENT 26 IS THE LOWER INTEGRATION     ZRUN0122
C     MINUS RUNGE-KUTTA INCREMENTS.  ERRORZ COMPUTES THE MAXIMUM        ZRUN0123
C     RELATIVE ERROR AND STAGEMENT 27 TESTS THIS ERROR AGAINST THE      ZRUN0124
C     LIMIT VALUE.                                                      ZRUN0125
   24 GO TO (25,28,28,30),I                                             ZRUN0126
   25 RATIO = DELT/OLDDEL                                               ZRUN0127
      HFACT=DELT/(1.+RATIO)                                             ZRUN0128
      ACOEF1=-RATIO*RATIO*HFACT                                         ZRUN0129
      ACOEF2=RATIO*(DELT+3.*OLDDEL)                                     ZRUN0130
      ACOEF3=DELT+DELT+HFACT                                            ZRUN0131
      DO 26 J=NST,NEQ                                                   ZRUN0132
   26 XINC(J) = ACOEF1*XDOTPM(J,1)+ACOEF2*XDOTPM(J,2)-6.*XINC(J)        ZRUN0133
     1+ACOEF3*XDOT(J)                                                   ZRUN0134
      CALL ERRORZ                                                       ZRUN0135
   27 IF (E2-ERLIMT) 28,28,34                                           ZRUN0136
C                                                                       ZRUN0137
C     PART 7. LAST POINT OKAY.  ADVANCE THE REMAINING PARAMETERS, FIND  ZRUN0138
C     NEW STEP SIZE.                                                    ZRUN0139
   28 H2 = DELT                                                         ZRUN0140
   29 OLDDEL = DELT                                                     ZRUN0141
   30 CALL STEP (OUTPUT)                                                ZRUN0142
      IF(DELT)31,42,31                                                  ZRUN0143
   31 DO 32 J = NST,NEQ                                                 ZRUN0144
      XDOTPM(J,1) = XDOTPM(J,2)                                         ZRUN0145
      XDOTPM(J,2) = XDOT(J)                                             ZRUN0146
      XPRIM(J,1)=XPRIM(J,2)                                             ZRUN0147
      XINC(J) = 0.                                                      ZRUN0148
   32 CONTINUE                                                          ZRUN0149
      GO TO (9,9,3,1),I                                                 ZRUN0150
C                                                                       ZRUN0151
C     PART 8. COMES HERE WHEN ERROR TEST FAILED--BOTH STARTING AND RUN. ZRUN0152
C     RETRIEVE OLD POINT AND RECOMPUTE WITH SMALLER INTERVAL.           ZRUN0153
C     IF TWO CONSECUTIVE TRYS FAIL (STATEMENT 36) THE STARTING SEQUENCE ZRUN0154
C     OCCURS.                                                           ZRUN0155
   33 ASSIGN 2 TO IBEGIN                                                ZRUN0156
   34 DO 35 J = NST,NEQ                                                 ZRUN0157
      XPRIM(J,2) = XPRIM(J,1)                                           ZRUN0158
      XDOT(J)=XDOTPM(J,2)                                               ZRUN0159
      XINC(J)= 0.                                                       ZRUN0160
   35 CONTINUE                                                          ZRUN0161
      STEPNO = STEPNO + 1                                               ZRUN0162
      H2 = DELT                                                         ZRUN0163
      DELT=SIGN (EXP ((ERLOG-A2)/5.),DELT)                              ZRUN0164
      A2 =A1                                                            ZRUN0165
   36 IF (FAIL-STEPGO) 37,38,37                                         ZRUN0166
   37 FAIL = STEPGO                                                     ZRUN0167
      GO TO IBEGIN, (9,2)                                               ZRUN0168
   38 ASSIGN 2 TO IBEGIN                                                ZRUN0169
      IF(STEPNO+STEPGO-STEPMX)39,39,40                                  ZRUN0170
   39 GO TO IBEGIN, (9,2)                                               ZRUN0171
   40 CALL OUTPUT(1)                                                    ZRUN0172
      WRITE (6,41)                                                      ZRUN0173
   41 FORMAT(1X,20HSTEPNO+STEPGO=STEPMX)                                ZRUN0174
   42 RETURN                                                            ZRUN0175
      END                                                               ZRUN0176
```

```
      FUNCTION SADDA(N,M)                                                ZSAD0001
C                                                                        ZSAD0002
C           SUBROUTINE SADDA SUMS UP THE DIFFERENCE (S(J,2)-S(J,1))      ZSAD0003
C     FOR PHASES N THROUGH M.                                            ZSAD0004
C                                                                        ZSAD0005
      COMMON /CSTAR/ CMA(1000),CMB(1000)                                 ZSAD0006
      DIMENSION S      (6   ,2   )                                       ZSAD0007
      EQUIVALENCE (S       ,CMB(074))                                    ZSAD0008
      SADDA=0.0                                                          ZSAD0009
      IF(N.GT.M) RETURN                                                  ZSAD0010
      DO 1 J = N,M                                                       ZSAD0011
    1 SADDA=SADDA+S(J,2)-S(J,1)                                          ZSAD0012
      RETURN                                                             ZSAD0013
      END                                                                ZSAD0014
```

```
      FUNCTION SADDB(I,N,M)                                              ZSAD0001
C                                                                        ZSAD0002
C           SUBROUTINE SADDB SUMS UP THE S(J,I) S FOR PHASES N           ZSAD0003
C     THROUGH M.                                                         ZSAD0004
      COMMON /CSTAR/ CMA(1000),CMB(1000)                                 ZSAD0005
      DIMENSION S      (6   ,2   )                                       ZSAD0006
      EQUIVALENCE (S       ,CMB(074))                                    ZSAD0007
      SADDB=0.0                                                          ZSAD0008
      IF(N.GT.M) RETURN                                                  ZSAD0009
      DO 1 J = N,M                                                       ZSAD0010
    1 SADDB=SADDB+S(J,I)                                                 ZSAD0011
      RETURN                                                             ZSAD0012
      END                                                                ZSAD0013
```

```
      SUBROUTINE SCOMP(N)                                               ZSC00001
C                                                                       ZSC00002
C          SUBROUTINE SCOMP COMPUTES VALUES TO WHICH THE                ZSC00003
C     DIFFERENCES IN FINAL ARE COMPARED IN ORDER TO CHECK THE           ZSC00004
C     RELATIVE MAGNITUDE OF THE ERROR IN FINAL CONDITIONS.              ZSC00005
C                                                                       ZSC00006
      COMMON /CSTAR/ CMA(1000),CMB(1000)                                ZSC00007
      COMMON /FINCMP/ FNCP(6,5)                                         ZSC00008
      DIMENSION COMP  (5      ),CONST (5   ,2   ),FLOMX (6      )       ZSC00009
      DIMENSION FYD   (5      ),IDATA (6   ,5   ),JFINAL(6      )       ZSC00010
      DIMENSION S     (6   ,2   ),X     (100    )                       ZSC00011
      EQUIVALENCE (BETA  ,CMA(880)),(COMP  ,CMB(031)),(CONST ,CMA(894)) ZSC00012
      EQUIVALENCE (ENERGY,CMA(892)),(FLOMX ,CMA(837)),(FM    ,CMA(715)) ZSC00013
      EQUIVALENCE (FYD   ,CMB(036)),(IDATA ,CMB(086)),(JFINAL,CMB(136)) ZSC00014
      EQUIVALENCE (LAST  ,CMA(890)),(NCUTE ,CMA(893)),(NFINAL,CMA(879)) ZSC00015
      EQUIVALENCE (NOPTA ,CMB(070)),(OMEGA ,CMA(405)),(R     ,CMA(402)) ZSC00016
      EQUIVALENCE (S     ,CMB(074)),(U     ,CMA(404)),(V     ,CMA(889)) ZSC00017
      EQUIVALENCE (X     ,CMA(401)),(ZLAM1 ,CMA(406)),(ZLAM2 ,CMA(407)) ZSC00018
      EQUIVALENCE (ZLAM3 ,CMA(408))                                     ZSC00019
      I = 0                                                             ZSC00020
      IF(N.GT.1) GO TO  4                                               ZSC00021
    1 IF(NFINAL.NE.1) RETURN                                            ZSC00022
C                                                                       ZSC00023
C     AS A CONVENIENCE FOR THE USER, RADIUS, FLIGHT PATH ANGLE, AND     ZSC00024
C     ENERGY MAY BE SUBSTITUTED FOR RADIUS, RADIAL VELOCITY, AND ANGULARZSC00025
C     VELOCITY BY SETTING THE ANGULAR VELOCITY DESIRED TO ZERO.  IF     ZSC00026
C     ANGULAR VELOCITY (FYD(3)) IS ZERO, THE RADIAL AND ANGULAR         ZSC00027
C     VELOCITIES ARE COMPUTED FROM THE FLIGHT PATH ANGLE AND ENERGY.    ZSC00028
    2 IF(FYD(3).NE.0.0) GO TO 3                                         ZSC00029
      VEL = SQRT(2.0*(ENERGY+FM/FYD(1)))                                ZSC00030
      FYD(2)=VEL*SIN (BETA)                                             ZSC00031
      FYD(3)=VEL*COS(BETA)/FYD(1)                                       ZSC00032
C                                                                       ZSC00033
C     COMPARISON VALUES FOR SPECIFIED RADIUS, RADIAL VELOCITY, AND      ZSC00034
C     ANGULAR VELOCITY                                                  ZSC00035
C                                                                       ZSC00036
    3 COMP(1)=1./FYD(1)                                                 ZSC00037
      COMP(2)=SQRT(FYD(1)/FM)                                           ZSC00038
      COMP(3)=COMP(2)*FYD(1)                                            ZSC00039
      RETURN                                                            ZSC00040
    4 IF(NCUTE.EQ.0) I = 1                                              ZSC00041
      GO TO (8,5,6,7),NFINAL                                            ZSC00042
C                                                                       ZSC00043
C     COMPARISON VALUES FOR SPECIFIED ENERGY, WITH OPTIMIZED            ZSC00044
C     RADIUS AND VELOCITY.                                              ZSC00045
    5 COMP(1)=1.0/ZLAM3/V                                               ZSC00046
      COMP(2)=1.0/ZLAM2/V                                               ZSC00047
      COMP(3)=R/FM                                                      ZSC00048
C                                                                       ZSC00049
      GO TO 8                                                           ZSC00050
C                                                                       ZSC00051
C     COMPARISON VALUES FOR SPECIFIED ENERGY AND FLIGHT PATH ANGLE      ZSC00052
C     WITH OPTIMIZED RADIUS.                                            ZSC00053
    6 COMP(1)=1.0                                                       ZSC00054
      COMP(2)=R/FM                                                      ZSC00055
      COMP(3)=1.0/ZLAM3                                                 ZSC00056
```

```
      GO TO 8
C
C     COMPARISON VALUES FOR SPECIFIED ENERGY AND PERIGEE RADIUS, WITH
C     OPTIMIZED INJECTION TRUE ANOMALY.
    7 COMP(1)=1.0/FYD(1)
      COMP(2)=1.0/SQRT((ZLAM1*(R*OMEGA**2-FM/R**2))**2+U**2*(ZLAM3**2
     1+4.0*OMEGA**2+ZLAM2**2))
      COMP(3)=R/FM
C
C     THIS SECTION CONTAINS THE COMPARISON VALUES FOR THE FINAL
C     CONDITIONS FOR OPTIMIZING PHASE DURATION.  THE EQUATION NUMBERS
C     FROM PAYLOAD OPTIMIZATON OF MULTISTAGE LAUNCH VEHICLES ARE LISTED
C     WITH THE EQUATION.
    8 DO 14 J = 1,LAST
      IF(JFINAL(J).EQ.0) GO TO 14
      K = JFINAL(J)
      I = I+1
      GO TO (9,12),K
    9 IF(IDATA(J+1,5).EQ.0) GO TO 10
C
C     EQUATION 41C
      COMP(I+2)=1.0/CONST(J,2)
C
      GO TO 14
   10 IF (J.GT.NOPTA+1) GO TO 11
C
C     EQUATION 48B
      COMP(I+2)=1.0/SQRT(FNCP(J,1)**2+FNCP(J,2)**2)
C
      GO TO 14
C
C     EQUATION 48A
   11 COMP(I+2)=1.0/SQRT(FNCP(J,1)**2+FNCP(J,2)**2+FNCP(J,3)**2
     1+FNCP(J,4)**2+FNCP(J,5)**2)
C
      GO TO 14
C
   12 IF(J.EQ.LAST) GO TO 13
C
C     EQUATION 47
      COMP(I+2)=1.0/SQRT(FNCP(J,1)**2*(FNCP(J,2)**2+FNCP(J,3)**2+
     1FNCP(J,4)**2)+FNCP(J,5)**2)
      GO TO 14
C     EQUATION 50
   13 COMP(I+2)=1.0/SQRT(FNCP(J,1)**2+FNCP(J,2)**2*(FNCP(J,3)**2
     1+FNCP(J,4)**2+FNCP(J,5)**2))
   14 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE SETUP
C
C          SUBROUTINE SETUP CONVERTS THE MODE INFORMATION WHICH
C     ENTERS THROUGH THE INPUT ROUTINE INTO MODOUT AND MODS WHICH
C     ARE USED IN STEP (SEE STEP).  ALSO TMIN AND STEPS ACQUIRE THE
C     PROPER VALUES FROM INPUT DATA.  THE INITIAL DELT IS COMPUTED
C     IN THIS ROUTINE.  THIS ROUTINE APPLIES TO THE BOOSTER PORTION
C     ONLY.
C
      COMMON /CSTAR/ CMA(1000),CMB(1000)
      DIMENSION DELMXB(2       ),MODEB (2       ),STEPB (2       )
      DIMENSION TMINB (2       ),TS    (6       )
      EQUIVALENCE (DELMAX,CMA(702)),(DELMXB,CMB(174)),(DELT  ,CMA(701))
      EQUIVALENCE (DELTBT,CMA(803)),(IMODE ,CMB(061)),(K     ,CMB(182))
      EQUIVALENCE (MODEB ,CMB(172)),(MODOUT,CMA(714)),(MODS  ,CMA(712))
      EQUIVALENCE (NSTAGE,CMA(710)),(STEPB ,CMB(176)),(STEPS ,CMA(704))
      EQUIVALENCE (TIME  ,CMA(017)),(TMIN  ,CMA(703)),(TMINB ,CMB(178))
      EQUIVALENCE (TS    ,CMA(932))
      INTEGER STEPB,STEPS
C     THIS ROUTINE IS SIMILAR TO DAMODE,BUT REFERS TO THE BOOSTER
      N=IMODE+1
C
C     K IS USED IN STEP AS AN IDEX OF NOUT
      K = 3
C
C     SET MODOUT
      MODOUT = MODEB(N)/10
C
C     SET MODS
      MODS = MOD(MODEB(N),10)
C
C     SET DELMAX
      DELMAX = DELMXB(N)
C
C     SET STEPS
      STEPS = STEPB(N)
C
C     SET TMIN
      TMIN = TMINB(N)
C
C     SET DELT
      DELT=AMIN1(DELTBT,TS(NSTAGE)-TIME)
      IF(DELT.GT.TMIN-TIME.AND.MODOUT.EQ.1) DELT = TMIN
      IF(DELT.GT.DELMAX.AND.(MODOUT.EQ.2.OR.MODOUT.EQ.3))DELT=DELMAX
C
      RETURN
      END
```

```
      SUBROUTINE SORTXY(X,Y,NPTS)
C
C          THIS ROUTINE IS CALLED BY MAINA TO ORDER THE LIST OF KICK
C     ANGLES.  IT WAS OBTAINED FROM REFERENCE 6.
C
      DIMENSION X(1),Y(1)
  100 N=NPTS
  102 NN=N-1
  104 DO 140 KT=1,NN
      XMIN=X(KT)
      JAD=KT
      JKL=KT+1
  112 DO 120 JK=JKL,N
  114 IF(XMIN-X(JK)) 120,120,116
  116 XMIN=X(JK)
  118 JAD=JK
  120 CONTINUE
  122 YMIN=Y(JAD)
      X(JAD)= X(KT)
      Y(JAD)= Y(KT)
      X(KT)= XMIN
      Y(KT)=YMIN
  140 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE SIMPRO
C
C          SUBROUTINE SIMPRO COMPUTES THE THRUST AND FLOW
C     FOR THE BOOSTER PORTION OF THE PROGRAM UTILIZING THRUST
C     ACTUAL = THRUST VACUUM -EXIT AREA * ATMOSPHERIC PRESSURE.
C
C
C
      COMMON /CSTAR/ CMA(1000),CMB(1000)
      DIMENSION EXITS (6       ),FORCES(6       ),WTFLOW(6       )
      EQUIVALENCE (EXITS ,CMA(727)),(FLOW   ,CMA(752)),(FORCES,CMA(739))
      EQUIVALENCE (NSTAGE,CMA(710)),(PA     ,CMA(806)),(PUSH   ,CMA(751))
      EQUIVALENCE (WTFLOW,CMA(733))
      PUSH = FORCES(NSTAGE)-PA*EXITS(NSTAGE)
      FLOW=WTFLOW(NSTAGE)
      CALL THRUST
      RETURN
      END
```

```
      FUNCTION SSTAGE(THRUST,FLOW,WT)                                   ZSST0001
C                                                                       ZSST0002
C          SUBROUTINE SSTAGE COMPUTES THE S S DEFINED IN EQUATION       ZSST0003
C     41 IN PAYLOAD OPTIMIZATION OF MULTISTAGE LAUNCH VEHICLES.         ZSST0004
C     AS DEFINED IN THE REPORT, THESE EQUATIONS ARE USED TO EFFECT      ZSST0005
C     OPTIMUM PHASING.                                                  ZSST0006
C                                                                       ZSST0007
      COMMON /CSTAR/ CMA(1000),CMB(1000)                                ZSST0008
      EQUIVALENCE (FM     ,CMA(715)),(G     ,CMA(716)),(OMEGA ,CMA(405)) ZSST0009
      EQUIVALENCE (R      ,CMA(402)),(U     ,CMA(404)),(ZLAM1 ,CMA(406)) ZSST0010
      EQUIVALENCE (ZLAM2 ,CMA(407)),(ZLAM3 ,CMA(408)),(ZLAM4 ,CMA(884)) ZSST0011
      IF (FLOW.EQ.0.0) GO TO 1                                          ZSST0012
      SSTAGE=((FM/R**2-OMEGA**2*R)*ZLAM1+2.0*U*OMEGA*ZLAM2-U*           ZSST0013
     1ZLAM3-OMEGA*ZLAM4-G*THRUST/WT*SQRT(ZLAM1**2+ZLAM2**2))/FLOW       ZSST0014
      GO TO 2                                                           ZSST0015
    1 SSTAGE=0.0                                                        ZSST0016
    2 RETURN                                                            ZSST0017
      END                                                               ZSST0018
```

```
      SUBROUTINE STAGE
C
C          SUBROUTINE STAGE IS CALLED FROM STEP TO PERFORM THE
C     OPERATIONS NEEDED TO CHANGE FROM ONE PHASE TO THE NEXT.
C
      COMMON /RUNG/RUN(125)
      COMMON /CSTAR/ CMA(1000),CMB(1000)
      DIMENSION COMPA (3      ),DRAG  (3      ),FLOMX (6      )
      DIMENSION FORCE (3      ),FUEL  (6      ),HARD  (6      )
      DIMENSION HARDB (6      ),OBLAT (3      ),PROP  (6      )
      DIMENSION S     (6  ,2  ),THRUST(6      ),VELEXP(6      )
      DIMENSION X     (100    ),XPRIM (100,2  )
      EQUIVALENCE (CD    ,CMA(812)),(COMPA ,CMA(783)),(CPSI  ,CMA(888))
      EQUIVALENCE (DELTAV,CMA(861)),(DRAG  ,CMA(777)),(DROP  ,CMA(863))
      EQUIVALENCE (FLOMX ,CMA(837)),(FLOW  ,CMA(877)),(FM    ,CMA(715))
      EQUIVALENCE (FORCE ,CMA(774)),(FUEL  ,CMA(871)),(FUELDV,CMA(878))
      EQUIVALENCE (G     ,CMA(716)),(HARD  ,CMA(843)),(HARDB ,CMA(721))
      EQUIVALENCE (IMODE ,CMB(061)),(JCOST ,CMB(129)),(JDATA ,CMA(925))
      EQUIVALENCE (LAST  ,CMA(890)),(NEQ   ,CMA(709)),(NSTAGE,CMA(710))
      EQUIVALENCE (NSTAG1,RUN(115)),(OBLAT ,CMA(780)),(OMEGA ,CMA(405))
      EQUIVALENCE (PA    ,CMA(806)),(PROP  ,CMA(849)),(QVAL  ,CMA(808))
      EQUIVALENCE (R     ,CMA(402)),(RESFRV,CMA(862)),(RMASS ,CMA(201))
      EQUIVALENCE (S     ,CMB(074)),(SPSI  ,CMA(887)),(THRUST,CMA(831))
      EQUIVALENCE (U     ,CMA(404)),(VELEX ,CMA(870)),(VELEXP,CMA(864))
      EQUIVALENCE (VMACH ,CMA(811)),(WEIGHT,CMA(203)),(X     ,CMA(401))
      EQUIVALENCE (XPRIM ,CMA(001))
      DOUBLE PRECISION XPRIM
C
C     IMODE = 2 IMPLIES UPPER PHASE OPERATION
      IF(IMODE.EQ.2) GO TO 2
C
C     REINITIALIZE
      CD=0.0
      VMACH=0.0
      PA=0.0
      QVAL=0.0
      DO 1 J = 1,3
      FORCE(J)=0.0
      DRAG(J)=0.0
      OBLAT(J)=0.0
    1 COMPA(J)=0.0
C
      IF(IMODE.EQ.0) RETURN
C     DROP WEIGHT BETWEEN SEGMENTS FOR BOOSTER OPERATION
      WEIGHT = WEIGHT-HARDB(NSTAG1)
      X(2)=WEIGHT
      RETURN
C
C     UPPER PHASE
C
C     CALL CONEVL TO EVALUATE CONSTANT
    2 CALL CONEVL(FLOW,NSTAG1-1)
C
C     COMPUTE S(N,2) BEFORE PHASING
      IF(FLOW.NE.0.0) GO TO 3
      S(NSTAG1,2)=0.0
```

```
      GO TO 4
    3 S(NSTAG1,2)=SSTAGE(THRUST(NSTAG1),FLOW,RMASS)
C
C     COMPUTE FUEL REQUIRED TO PERFORM IDEAL DELTAV AFTER
C     LAST PHASE.
    4 IF(NSTAGE.NE.LAST+1) GO TO 6
      IF(VELEXP(LAST).EQ.0.0) GO TO 5
      FUELDV=(RMASS-DROP)*(1.0-EXP(-(DELTAV+RESERV)/VELEXP(LAST)))
      RETURN
    5 FUELDV=0.0
      RETURN
C
C     DROP WEIGHT BETWEEN PHASES
    6 RMASS=RMASS-HARD(NSTAG1)-FUEL(NSTAG1)*PROP(NSTAG1)
C
      X(1)=RMASS
C
C     SET JET VELOCITY AND FLOW FOR NEXT PHASE
      VELEX=VELEXP(NSTAGE)
      FLOW=FLOMX(NSTAGE)
C
C     COMPUTE S FOR BEGINNING OF NEW PHASE
      S(NSTAGE,1)=SSTAGE(THRUST(NSTAGE),FLOW,RMASS)
C
C     SET FLAG FOR COAST
      JCOST=6
C
      RETURN
      END
```

```
      SUBROUTINE START(N)                                               ZSTA0001
C                                                                       ZSTA0002
C          SUBROUTINE START PERFORMS THE REQUIRED INITIALIZATIONS       ZSTA0003
C     BEFORE RUNGEK IS CALLED TO PERFORM THE INTEGRATION.               ZSTA0004
C                                                                       ZSTA0005
      COMMON/ATABLE/CME(8000)                                           ZSTA0006
      COMMON /CSTAR/ CMA(1000),CMB(1000)                                ZSTA0007
      DIMENSION FLOMX (6       ),FUEL   (6       ),HARD   (6       )    ZSTA0008
      DIMENSION IDATA (6   ,5  ),JCOAST(6       ),NOPT   (6       )    ZSTA0009
      DIMENSION NTB   (4       ),PAR    (6   ,2  ),PROP   (6       )    ZSTA0010
      DIMENSION S     (6   ,2  ),TB     (6       ),TBOOST(6       )    ZSTA0011
      DIMENSION TEMP  (10      ),THRUST(6       ),TS     (6       )    ZSTA0012
      DIMENSION V     (6       ),VELEXP(6       ),XINPT  (100     )    ZSTA0013
      DIMENSION XO    (6   ,5  )                                        ZSTA0014
      EQUIVALENCE (ELEV  ,CMA(790)),(FIXDTK,CMB(071)),(FLOMX ,CMA(837)) ZSTA0015
      EQUIVALENCE (FLOW  ,CMA(877)),(FM    ,CMA(715)),(FUEL  ,CMA(871)) ZSTA0016
      EQUIVALENCE (FUELT ,CME(014)),(G     ,CMA(716)),(HARD  ,CMA(843)) ZSTA0017
      EQUIVALENCE (HARDBT,CME(015)),(IDATA ,CMB(086)),(IKICK ,CME(200)) ZSTA0018
      EQUIVALENCE (IMODE ,CMB(061)),(ITER  ,CMB(068)),(ITERAD,CMB(067)) ZSTA0019
      EQUIVALENCE (ITERPD,CMB(060)),(JCOAST,CMB(130)),(JCOST ,CMB(129)) ZSTA0020
      EQUIVALENCE (LAST  ,CMA(711)),(LAST1 ,CMA(753)),(LAST2 ,CMA(890)) ZSTA0021
      EQUIVALENCE (MASH  ,CMB(064)),(NFO   ,CMA(709)),(NOPT  ,CMA(819)) ZSTA0022
      EQUIVALENCE (NSTAGE,CMA(710)),(NTB   ,CMB(116)),(OMEGA ,CMA(605)) ZSTA0023
      EQUIVALENCE (PAR   ,CMB(142)),(PROP  ,CMA(849)),(PSIDO ,CMA(882)) ZSTA0024
      EQUIVALENCE (PSIO  ,CMA(881)),(R     ,CMA(602)),(ROA   ,CMB(062)) ZSTA0025
      EQUIVALENCE (S     ,CMB(074)),(TB    ,CMA(825)),(TBOOST,CMA(745)) ZSTA0026
      EQUIVALENCE (TBURN ,CMB(065)),(THRUST,CMA(831)),(TKICK ,CMB(059)) ZSTA0027
      EQUIVALENCE (TS    ,CMA(932)),(TSPM  ,CME(010)),(U     ,CMA(604)) ZSTA0028
      EQUIVALENCE (V     ,CMB(154)),(VELEX ,CMA(870)),(VELEXP,CMA(864)) ZSTA0029
      EQUIVALENCE (WTO   ,CMA(720)),(XINPT ,CMA(601)),(XO    ,CMB(001)) ZSTA0030
      EQUIVALENCE (ZLAMO ,CMA(883)),(ZLAM1 ,CMA(606)),(ZLAM2 ,CMA(607)) ZSTA0031
      EQUIVALENCE (ZLAM3 ,CMA(608)),(ZLAM4 ,CMA(884))                   ZSTA0032
C     ITERPD=0      OPTIMIZE  TKICK                                     ZSTA0033
C     ITERPD=1      FIXED     TKICK                                     ZSTA0034
      DATA IHUNT/0/                                                     ZSTA0035
      INTEGER FIXDTK                                                    ZSTA0036
C                                                                       ZSTA0037
C                                                                       ZSTA0038
C     SET TIME RATE OF CHANGE OF THRUST ANGLE FROM PERTURBATION         ZSTA0039
C     ARRAY (XO)                                                        ZSTA0040
      PSIDO=XO(N,1)                                                     ZSTA0041
C                                                                       ZSTA0042
C     ITER EQUALS NUMBER OF PHASE DURATIONS IN EXTERNAL ITERATION.      ZSTA0043
      IF(ITER.EQ.0) GO TO 2                                             ZSTA0044
C                                                                       ZSTA0045
      KB=2                                                              ZSTA0046
C                                                                       ZSTA0047
C     ITERAD EQUAL TO THREE IMPLIES THAT THE BOOSTER BURNING TIME       ZSTA0048
C     IS PART OF THE ITERATION.                                         ZSTA0049
      IF(ITERAD.EQ.3) KB=1                                              ZSTA0050
C                                                                       ZSTA0051
      DO 1 J = 1,ITER                                                   ZSTA0052
C                                                                       ZSTA0053
C     NTB(J) CONTAINS THE PHASE NUMBERS OF THE PHASE DURATIONS TO BE    ZSTA0054
C     OPTIMIZED IN THE EXTERNAL ITERATION.                              ZSTA0055
      K = NTB(J)                                                        ZSTA0056
```

```
C
      KA=J+KB
C
C     SET PHASE DURATION FOR APPROPRIATE PHASES FROM PERTURBATION
C     ARRAY (XO)
    1 TB(K)=XO(N,KA)
C
    2 IF(ITERAD.EQ.3) GO TO 4
C
C     ITERPD EQUAL TO ONE IMPLIES THAT THE KICK ANGLE FOR THE BOOSTER
C     IS FIXED AND THE THRUST ANGLE (PSIO) MUST BE SET FROM THE
C     PERTURBATION ARRAY (XO)
      IF(ITERPD.EQ.1) GO TO 3
C
C     SET KICK ANGLE FROM PERTURBATION ARRAY (XO)
      TKICK=XO(N,2)
C
      GO TO 4
    3 PSIO=XO(N,2)
C
C     INTEGRATE BOOSTER FOR FIXED KICK ANGLE AND BURNING TIME
    4 GO TO (5,8,6),FIXDTK
C
C     SET IMODE FOR BOOSTER
    5 IF(IKICK.NE.0.AND.IMODE.NE.0) GO TO 51
C     INTEGRATE VERTICAL RISE IF REQUIRED
      IMODE=0
      CALL MAINB
   51 IMODE=1
      TBURN=TB(1)+TSPM
C
C     SETUP SETS OUTPUT CONTROLS
      CALL SETUP
C
C     SET ELEVATION ANGLE EQUAL TO TKICK
      ELEV=TKICK
C
C     SET TIME FOR END OF BOOSTER PORTION
      TBOOST(LAST1)=TB(1)
C
C     PRINT HEADINGS
      CALL OUTPT2(0)
C
C     MAINB CALLS RUNGEK TO INTEGRATE TRAJECTORY.
      CALL MAINB
C
C     SET FIXDTK TO TWO TO PREVENT REINTEGRATION OF THE BOOSTER.
      FIXDTK=2
C
      IMODE=2
      LAST=LAST2
      GO TO 7
C
C     MAINA CONTROLS GENERATION OF BOOSTER TABLE AND INTERPOLATION
C     SCHEME.
    6 CALL MAINA
C
C     IF MASH EQUALS ONE, DIFFICULTIES HAVE BEEN ENCOUNTERED IN MAINA
C     AND CONTROL IS RETURNED TO MAIN FOR CORRECTION.
      IF (MASH.EQ.1) RETURN
```

```
C
C     COMPUTE INITIAL WEIGHT OF FIRST UPPER PHASE
    7 XINPT(1)=WTO-FUELT-HARDBT-(1.0+PROP(1))*FLOMX(1)*TB(1)-HARD(1)
C
C     COMPUTE RADIUS
      XINPT(2)=ROA+V(1)
C     SET INITIAL RADIAL VELOCITY
      XINPT(4)=V(2)
C     COMPUTE INITIAL ANGULAR VELOCITY
      XINPT(5)=V(3)/XINPT(2)
C     SET TIME AT BEGINNING FOR FIRST UPPER PHASE
      XINPT(9)=TBURN
C
C     SET TS(1) TO TOTAL BURNING TIME OF BOOSTER SEGMENTS
    8 TS(1)=TBURN
C
      LAST=LAST2
C
C     SET FLAG FOR COAST
      JCOST=6
C
C     COMPUTE PROPELLANT LOADING FOR ALL UPPER PHASES FOR
C     WHICH THE PHASE DURATION HAS BEEN DETERMINED AND SET THE
C     TS LIST FOR AS MANY AS POSSIBLE (UNTIL A PHASE IS ENCOUNTERED
C     WHICH IS TERMINATED BY COAST)
      FUEL(1)=TB(1)*FLOMX(1)
      DO 9   J = 2,LAST
      IF(JCOAST(J).NE.0) GO TO 10
      FUEL(J)=TB(J)*FLOMX(J)
    9 TS(J)=TS(J-1)+TB(J)
      GO TO 11
   10 TS(J)=TS(J-1)+10000.0
C
C     SET NSTAGE FOR UPPER PHASES
   11 NSTAGE=2
C
C     SET JET VELOCITY AND FLOW RATE FOR THE FIRST UPPER PHASE
      VELEX=VELEXP(2)
      FLOW=FLOMX(2)
C
      GO TO (12,16,16),ITERAD
   12 IF(IDATA(2,5).EQ.1) GO TO 13
C
C     EQUATIONS B10 AND B13
      APART=(PAR(2,1)/FLOMX(1)+(1.0+PROP(1))/FLOMX(2)*(FM/R**2-OMEGA**2*
     1R))*PAR(1,2)-(1.0/FLOMX(1)-(1.0+PROP(1))/FLOMX(2))*U*PAR(2,2)
      BPART=(R/FLOMX(1)*PAR(3,1)+2.0*(1.0+PROP(1))/FLOMX(2)*U*OMEGA)
     1*PAR(1,2)-(1.0/FLOMX(1)-(1.0+PROP(1))/FLOMX(2))*U*R*PAR(3,2)
      CPART=(1.0/FLOMX(1)-(1.0+PROP(1))/FLOMX(2))*(OMEGA)*PAR(1,2) *
     1ZLAM4+ZLAMO*G*THRUST(2)/XINPT(1)*(1.0+PROP(1))/FLOMX(2)*PAR(1,2)
      GO TO 14
C     EQUATIONS B10 AND B13 WHEN FLOW = 0
C     EQUIATIONS B10 AND B13 WHEN FLOW = 0
   13 APART=(FM/R**2-OMEGA**2*R)*PAR(1,2)+U*PAR(2,2)
      BPART=2.0*U*OMEGA*PAR(1,2)+U*R*PAR(3,2)
      CPART=(OMEGA*PAR(1,2))*ZLAM4
   14 ABSQ=APART**2+BPART**2
C
C     EQUATION B12A
      ZLAM1=(APART*CPART-BPART*SQRT(ABSQ*ZLAMO**2-CPART**2))/ABSQ
```

```
C
C     EQUATION B12B
      ZLAM2=(BPART*CPART+APART*SQRT(ABSQ*ZLAMO**2-CPART**2))/ABSQ
C
      PSIO=ARCTAN(ZLAM1,ZLAM2)
      ZLAM3=ZLAMO**2/ZLAM2*(2.0*OMEGA-PSIDO) -U/R*ZLAM1+ZLAM1*ZLAM4/R/
     1ZLAM2
      CALL ITERAT(TB(1),1.0E-02,ZLAM1*PAR(2,2)+R*ZLAM2*PAR(3,2)+ZLAM3
     1*PAR(1,2),0.0,5.0E-08,IHUNT,10,IPICK,2,TEMP,SQRT((ZLAM1*PAR(1,2))
     1**2+(ZLAM2*PAR(3,2)*R)**2))
      TBURN=TB(1)+TSPM
      GO TO (6,18,15),IPICK
   15 MASH=1
      RETURN
   16 SPSI=SIN(PSIO)
      CPSI=COS(PSIO)
      TPSI=SPSI/CPSI
      ZLAM1=SPSI*ZLAMO
      ZLAM2=CPSI*ZLAMO
C
C     EQUATION B6
      ZLAM3=ZLAMO*(2.0*OMEGA-PSIDO-U/R*SPSI*CPSI)/CPSI+ZLAM4/R*TPSI
      IF(ITERPD.EQ.1.AND.ITERAD.NE.3) GO TO 18
      IF(ITERAD.EQ.3) GO TO 17
C
C     EQUATION 40D
      CALL ITERAT(PSIO,1.0E-02,ZLAM1*PAR(2,2)+R*ZLAM2*PAR(3,2)+ZLAM3*
     1PAR(1,2),0.0,1.0E-06,IHUNT,10,IPICK,2,TEMP,SQRT((ZLAM1*PAR(2,2))
     2**2+(R*ZLAM2*PAR(3,2))**2))
C
      GO TO (16,18,15),IPICK
C
C     EQUATIONS B9 AND B10
   17 CALL ITERAT(PSIO,1.0E-02,(FLOMX(2)/FLOMX(1)*PAR(2,1)+(1.0+PROP(1)
     1)*(FM/R**2-OMEGA**2*R))*ZLAM1+(FLOMX(2)/FLOMX(1)*R*PAR(3,1)+
     22.0*(1.0+PROP(1))*U*OMEGA)*ZLAM2+U*(FLOMX(2)/FLOMX(1)-(1.0+PROP(1)
     3))*ZLAM3+(FLOMX(2)/FLOMX(1)-(1.0+PROP(1)))*OMEGA*ZLAM4-(1.0+
     4PROP(1))*G*THRUST(2)/XINPT(1)*ZLAMO,0.0,1.0E-06,IHUNT,10,IPICK,2,
     5TEMP,SQRT((FLOMX(2)/FLOMX(1)*PAR(2,1)+(1.0+PROP(1))*(FM/R**2-OM
     6EGA**2*R))**2*ZLAM1**2+(R*FLOMX(2)/FLOMX(1)*PAR(3,1)+2.0*(1.0+
     7PROP(1))*U*OMEGA)**2*ZLAM2**2))
      GO TO (16,18,15),IPICK
C
C     COMPUTE S REQUIRED FOR END OF BOOSTER PORTION
   18 S(1,2)=-(ZLAM1*PAR(2,1)+R*ZLAM2*PAR(3,1)+U*ZLAM3+OMEGA*ZLAM4)/
     1FLOMX(1)
C
C     COMPUTE S REQUIRED FOR BEGINNING OF UPPER PHASES
      IF(FLOMX(2).EQ.0.0) GO TO 19
      S(2,1)=((FM/R**2-OMEGA**2*R)*ZLAM1+2.0*U*OMEGA*ZLAM2-U*ZLAM3-OMEGA
     1*ZLAM4-THRUST(2)*G/XINPT(1)*SQRT(ZLAM1**2+ZLAM2**2))/FLOMX(2)
      RETURN
   19 S(2,1)=0.0
C
      RETURN
      END
```

```
      SUBROUTINE STDATA                                                 ZSTD0001
C                                                                       ZSTD0002
C     SUBROUTINE STDATA INITIALIZED THE DX ARRAY WHICH CONSISTS         ZSTD0003
C     OF THE PERTURBATION FACTORS.                                      ZSTD0004
C                                                                       ZSTD0005
      COMMON /CSTAR/ CMA(1000),CMB(1000)                                ZSTD0006
      DIMENSION DX     (5       )                                       ZSTD0007
      EQUIVALENCE (DX     ,CMA(938))                                    ZSTD0008
      DX(1)=1.0E-04                                                     ZSTD0009
      DX(2)=1.0E-04                                                     ZSTD0010
      DX(3)=1.0E-04                                                     ZSTD0011
      DX(4)=1.0E-04                                                     ZSTD0012
      DX(5)=1.0E-04                                                     ZSTD0013
      RETURN                                                            ZSTD0014
      END                                                               ZSTD0015
```

```
$*                                                                      ZSTD0016
$IBFTC ZSTEP   LIST,DECK                                                ZSTE0000
      SUBROUTINE STEP (OUTPUT)                                          ZSTE0001
C                                                                       ZSTE0002
C          SUBROUTINE STEP IS CALLED FROM RUNGEK TO CONTROL INTEGRATION ZSTE0003
C     STEPSIZE AND PRINTOUT.                                            ZSTE0004
C                                                                       ZSTE0005
C                                                                       ZSTE0006
C                                                                       ZSTE0007
C     MODOUT=1  OUTPUT EVERY NTH STEP(N=STEPS) UNTIL TIME = TMIN, THEN  ZSTE0008
C                  GO TO MODE 2 .                                       ZSTE0009
C            2  OUTPUT AT INTERVALS OF DELMAX UNTIL TIME = TMAX.        ZSTE0010
C            3  OUTPUT AT INTERVALS OF DELMAX UNTIL TIME = TMIN, THEN   ZSTE0011
C                  GO TO MODE 4 .                                       ZSTE0012
C            4  OUTPUT EVERY NTH STEP UNTIL TIME = TMAX.                 ZSTE0013
C            5  OUTPUT FIRST AND LAST STEP                              ZSTE0014
C            6  NO OUTPUT                                               ZSTE0015
C                                                                       ZSTE0016
C                                                                       ZSTE0017
      COMMON /RUNG/RUN(125)                                             ZSTE0018
      COMMON /CSTAR/ CMA(1000),CMB(1000)                                ZSTE0019
      DIMENSION NOUT  (6   ,3   ),TS    (6       ),XDOT  (100      )    ZSTE0020
      DIMENSION XPRIM (100,2   )                                        ZSTE0021
      EQUIVALENCE (A1    ,RUN(101)),(A2    ,RUN(102)),(ALT   ,CMA(805)) ZSTE0022
      EQUIVALENCE (DELMAX,CMA(702)),(DELSTO,RUN(104)),(DELT  ,CMA(701)) ZSTE0023
      EQUIVALENCE (DEL1  ,RUN(103)),(ERLIMT,CMA(706)),(ERLOG ,CMA(707)) ZSTE0024
      EQUIVALENCE (E2    ,RUN(105)),(H2    ,RUN(106)),(I     ,RUN(107)) ZSTE0025
      EQUIVALENCE (IBURN ,CMB(072)),(IMODE ,CMB(061)),(K     ,CMB(182)) ZSTE0026
      EQUIVALENCE (LAST  ,CMA(711)),(MASH  ,CMB(064)),(MODOUT,CMA(714)) ZSTE0027
      EQUIVALENCE (MODS  ,CMA(712)),(NEO   ,CMA(709)),(NOUT  ,CMB(183)) ZSTE0028
      EQUIVALENCE (NST   ,CMA(708)),(NSTAGE,CMA(710)),(NSTAG1,RUN(115)) ZSTE0029
      EQUIVALENCE (NSTEP1,RUN(108)),(NSTEP2,RUN(109)),(NSTEP3,RUN(110)) ZSTE0030
      EQUIVALENCE (RATIO ,RUN(111)),(SCRIBE,RUN(114)),(STEPGO,RUN(112)) ZSTE0031
      EQUIVALENCE (STEPMX,CMA(705)),(STEPNO,RUN(113)),(STEPS ,CMA(704)) ZSTE0032
      EQUIVALENCE (TIME  ,CMA(409)),(TMIN  ,CMA(703)),(TS    ,CMA(932)) ZSTE0033
      EQUIVALENCE (X     ,CMA(401)),(XDOT  ,CMA(501)),(XINC  ,RUN(001)) ZSTE0034
      EQUIVALENCE (XPRIM ,CMA(001))                                     ZSTE0035
      DOUBLE PRECISION XPRIM                                            ZSTE0036
      INTEGER STEPGO,STEPNO,SCRIBE,STEPS,STEPMX,FAIL                    ZSTE0037
C     CHECK FOR MAXIMUM NUMBER OF STEPS                                 ZSTE0038
      STEPGO = STEPGO + 1                                               ZSTE0039
      IF(IMODE.EQ.1.AND.ALT.LT.10.0) GO TO 32                           ZSTE0040
      IF(STEPGO+STEPNO.LT.STEPMX) GOTO 2                                ZSTE0041
      CALL OUTPUT(1)                                                    ZSTE0042
      WRITE (6,1)                                                       ZSTE0043
    1 FORMAT(1X,20HSTEPNO+STEPGO=STEPMX)                                ZSTE0044
      MASH = 1                                                          ZSTE0045
      GO TO 32                                                          ZSTE0046
    2 IF(IMODE.EQ.2) CALL COAST                                         ZSTE0047
      NSTAG1=NSTAGE                                                     ZSTE0048
      GO TO (3,34,34,4),I                                               ZSTE0049
    3 A3 = (A2-A1)*RATIO+A2                                             ZSTE0050
      DELT = SIGN (EXP ((ERLOG-A3)/5.),DELT)                            ZSTE0051
      IF(DELT/H2.GT.3.0) DELT=3.0*H2                                    ZSTE0052
    4 GO TO (5,7,6,8,15,15),MODOUT                                      ZSTE0053
C                                                                       ZSTE0054
C     CHECKS PROXIMITIY TO TMIN FOR CHANGE FROM MODOUT = 1 TO           ZSTE0055
C     MODOUT = 2.                                                       ZSTE0056
    5 DEL1 = TMIN-TIME                                                  ZSTE0057
C                                                                       ZSTE0058
```

```
      IF(ABS(DEL1/TMIN).GT.1.E-07) GO TO 8
C
C     SETS UP DEL1 S FOR LANDING ON TMIN AND CHANGES MODOUT FROM
C     ONE TO TWO
      MODOUT = 2
      GO TO 15
C
C     CHECK FOR PROXIMITY TO TMIN FOR CHANGING FROM MODOUT = 3
C     TO MODOUT = 4.
C
    6 IF(DELT*(TIME-TMIN).LE.0.0) GO TO 7
      MODOUT = 4
      NSTEP2=2
      GO TO 8
C
C     CALCULATES DEL FOR DELMAX PRINT OUT
    7 DEL1=DEL1-H2
      GO TO 15
C
C     CHECK FOR N-STEP PRINT OUT
    8 DEL3=FLOAT(MOD(STEPGO,STEPS))
      IF(DEL3.NE.0.0) GO TO 15
      CALL OUTPUT(1)
      GO TO 15
C
C     SPACES CALCULATES NUMBER OF STEPS TO PRINT OUT OR STAGING POINT
C
    9 SPACES =AINT ((DEL/DELT)+SIGN (.9,(DEL/DELT)))
C
C     SPACES = 0 INDICATES THAT A STAGING POINT OR PRINT OUT POINT
C     HAS BEEN REACHED.
C
      IF(SPACES.NE.0.0)     GO TO 23
      NSTEP1=2
      GO TO (14,10,10,14,14,14),MODOUT
C
C           DEL2 LESS THAN 0.                PRINT OUT ONLY
C           DEL2 = 0.                        BOTH STAGING AND PRINT
C                                            OUT POINT INDICATED
C           DEL2 GREATER THAN 0.             STAGING POINT ONLY
C
   10 DEL3=DEL2
      IF(DEL2)12,11,14
   11 NSTEP3=1
   12 IF(IMODE.EQ.1) IBURN=IBURN+1
      CALL OUTPUT(1)
      DEL1=DELMAX+DELSTO
      DELSTO=0.0
      GO TO (13,15),NSTEP3
   13 NSTEP3=2
C
C     NSTAGE IS INCREMENTED, I = 4 CAUSES RESTART IN RUNGEK
   14 NSTAGE=NSTAGE+1
      I=4
C
C     CALCULATION OF DEL FOR STAGING
   15 DEL=TS(NSTAGE)-TIME
      GO TO (16,18),NSTEP2
C
C     IF DELMAX PRINT OUT DESIRED, CALCULATES DEL FOR LANDING ON
```

```
C     NEXT SPACES = 0 POINT.                                            ZSTE0119
   16 DEL2=DEL1-DEL                                                     ZSTE0120
      IF(ABS(DEL2/TIME).GT..0000001)      GO TO 17                      ZSTE0121
      DELSTO=DEL2                                                       ZSTE0122
      DEL2=0.0                                                          ZSTE0123
      GO TO 18                                                          ZSTE0124
   17 DEL=SIGN(AMIN1 (ABS (DEL),ABS (DEL1)),DELT)                       ZSTE0125
   18 GO TO (9,20),NSTEP1                                               ZSTE0126
   20 NSTEP1=1                                                          ZSTE0127
      IF(ABS(DEL).GE.ABS(DELT)) GO TO 22                                ZSTE0128
C                                                                       ZSTE0129
C     NSTAGE IS INCREMENTED UNTIL TS(NSTAGE1+1) DOES NOT EQUAL          ZSTE0130
C     TS(NSTAGE)                                                        ZSTE0131
      IF (NSTAG1.EQ.NSTAGE.OR.TS(NSTAGE-1).NE.TS(NSTAGE)) GO TO 21      ZSTE0132
      NSTEP1=2                                                          ZSTE0133
      GO TO 14                                                          ZSTE0134
   21 DELT=SIGN(DEL,DELT)                                               ZSTE0135
C                                                                       ZSTE0136
   22 IF(I-4)9,24,9                                                     ZSTE0137
   23 DELT = DEL/SPACES                                                 ZSTE0138
      RETURN                                                            ZSTE0139
C                                                                       ZSTE0140
C                                                                       ZSTE0141
C     MODS = 1  NO PRINT BEFORE AND AFTER STAGING                       ZSTE0142
C            2  PRINT BEFORE AND AFTER STAGING                          ZSTE0143
C            3  PRINT BEFORE, NOT AFTER STAGING                         ZSTE0144
C            4  PRINT AFTER, NOT BEFORE STAGING                         ZSTE0145
C                                                                       ZSTE0146
C                                                                       ZSTE0147
   24 IF(NOUT(NSTAG1,K).NE.0) GO TO 28                                  ZSTE0148
      IF(MODOUT-5) 25,26,28                                             ZSTE0149
   25 IF(DEL3.EQ.0.0) GO TO 28                                          ZSTE0150
   26 GO TO (28,27,27,28),MODS                                          ZSTE0151
   27 CALL OUTPUT(1)                                                    ZSTE0152
   28 CALL STAGE                                                        ZSTE0153
      IF (MODOUT.EQ.6.OR.NOUT(NSTAG1,K).NE.0) GO TO 31                  ZSTE0154
      IF (NSTAGE.GT.LAST) GO TO 30                                      ZSTE0155
      GO TO (31,29,31,29),MODS                                          ZSTE0156
   29 SCRIBE = 0                                                        ZSTE0157
      GO TO 31                                                          ZSTE0158
   30 NSTAG1=NSTAGE                                                     ZSTE0159
      CALL OUTPUT(1)                                                    ZSTE0160
   31 IF(NSTAGE.LE.LAST) GO TO 9                                        ZSTE0161
   32 DELT=0.0                                                          ZSTE0162
C                                                                       ZSTE0163
C     RESET XPRIMS TO ZERO                                              ZSTE0164
C                                                                       ZSTE0165
      DO 33 J = 1,10                                                    ZSTE0166
   33 XPRIM(J,1)=0.0D0                                                  ZSTE0167
   34 RETURN                                                            ZSTE0168
      END                                                               ZSTE0169


      SUBROUTINE STGSS(KMAX)                                            ZSTG0001
C                                                                       ZSTG0002
C          SUBROUTINE STGSS IS A DUMMY ROUTINE PLACED IN THE            ZSTG0003
C     MAIN PROGRAM AFTER THE COMPLETION OF THE PROBLEM.                 ZSTG0004
C                                                                       ZSTG0005
      RETURN                                                            ZSTG0006
      END                                                               ZSTG0007
```

```
                        SUBROUTINE THRUST
C
C           SUBROUTINE THRUST SUPPLIES THE THRUST DIRECTION FOR
C     BOTH THE VERTICAL RISE PORTION AND ZERO ANGLE-OF-ATTACK
C     PORTION OF THE BOOSTER PHASE AND THE ACCELERATIONS FOR
C     BOTH OF THESE PORTIONS.
C
      COMMON /ATABLE/CME(8000)
      COMMON /CSTAR/ CMA(1000),CMB(1000)
      DIMENSION COMPA (3      ),DRAG   (3      ),FORCE (3      )
      DIMENSION H     (5      ),OBLAT  (3      ),RB    (5      )
      DIMENSION UF    (5      ),UH     (5      ),UN    (3      )
      DIMENSION UT    (3      ),UVATM  (5      ),VATM  (5      )
      EQUIVALENCE (COMPA ,CMA(783)),(DRAG  ,CMA(777)),(FORCE ,CMA(774))
      EQUIVALENCE (G     ,CMA(716)),(IMODE ,CMB(061)),(OBLAT ,CMA(780))
      EQUIVALENCE (PUSH  ,CMA(751)),(RB    ,CMA(754)),(UN    ,CME(022))
      EQUIVALENCE (UT    ,CMA(800)),(VATM  ,CMA(764)),(WEIGHT,CMA(402))
C
      TDPMAG=PUSH/WEIGHT*G
      IF(IMODE.EQ.1) GO TO 2
C
C     VERTICAL RISE SECTION
C     THE THRUST IS ALLIGNED PARALLEL TO UT WHERE UT IS A UNIT
C     VECTOR IN THE DIRECTION OF THE LAUNCH RADIUS VECTOR.
      DO 1  J=1,3
    1 FORCE(J)=TDPMAG*UT(J)
C
      GO TO 5
C     ZERO ANGLE-OF-ATTACK SECTION
C
C     DURING THIS PORTION OF THE BOOSTER, THE THRUST VECTOR IS
C     ALLIGNED SUCH THAT IT IS PARALLEL TO THE AZIMUTH PLANE
C     AND THE ANGLE BETWEEN THE RELATIVE VELOCITY VECTOR AND THE
C     PROJECTION OF THE THRUST VECTOR INTO THE RADIUS-RELATIVE
C     VELOCITY PLANE IS ZERO.
    2 CALL CONVT (RB,VATM,H)
      DO 3 J = 1,3
      UH(J)=H(J)/H(5)
    3 UVATM(J)=VATM(J)/VATM(5)
      UHDOTW=DOT(UH,UN)
      UVATMW=DOT(UVATM,UN)
      DENOM=SQRT(UVATMW**2+UHDOTW**2)
      DO 4 J = 1,3
      UF(J)=(UHDOTW*UVATM(J)-UVATMW*UH(J))/DENOM
    4 FORCE(J)=TDPMAG*UF(J)
    5 DO 6 J=1,3
    6 COMPA(J)=OBLAT(J)+FORCE(J)+DRAG(J)
      RETURN
      END
```

```
      SUBROUTINE TUDES
C
C          SUBROUTINE TUDES CONVERTS LATITUDE, LONGITUDE, AZIMUTH,
C     ELEVATION ANGLE, AND RELATIVE VELOCITY INTO RECTANGULAR
C     COORDINATES.
C
      COMMON /CSTAR/ CMA(1000),CMB(1000)
      COMMON /ATABLE/ CME(8000)
      DIMENSION ANGLEB(4      ),ANGLES(4      ),COSA  (4      )
      DIMENSION SINA  (4      ),UN    (3      ),UT    (3      )
      DIMENSION XPRIM (100,2  )
      EQUIVALENCE (A     ,CME(004)),(ALT   ,CME(005)),(ANGLEB,CME(025))
      EQUIVALENCE (ANGLES,CMA(786)),(B     ,CMA(814)),(ELEV  ,CMA(790))
      EQUIVALENCE (GM    ,CMA(715)),(IMODE ,CMB(061)),(OBLATJ,CMA(818))
      EQUIVALENCE (OBLATN,CMA(815)),(RADIUS,CME(013)),(REVOLV,CMA(799))
      EQUIVALENCE (RMASS ,CMA(003)),(SINA  ,CMA(791)),(TKTIME,CMA(804))
      EQUIVALENCE (UN    ,CME(022)),(UT    ,CMA(800)),(VEL   ,CME(001))
      EQUIVALENCE (XPRIM ,CMA(001))
      DOUBLE PRECISION  XPRIM
      DATA RADDEG/57.2957795/
      VEL1=VEL
      IF(IMODE.NE.0) GO TO 2
      DO 1 I = 1,4
C
C     ANGLES(1)           ANGLEB(1)           LATITUDE
C     ANGLES(2)           ANGLEB(2)           LONGITUDE
C     ANGLES(3)           ANGLEB(3)           AZIMUTH HEADING
C     ANGLES(4)           ANGLEB(4)           ELEVATION ANGLE
C
    1 ANGLEB(I)=ANGLES(I)
    2 DO 3 I = 1,4
      SINA(I)=SIN(ANGLEB(I)/RADDEG)
    3 COSA(I)=COS(ANGLEB(I)/RADDEG)
C
C     COMPUTE X,Y,Z COMPONENTS
      XPRIM(6,1)=DBLE(COSA(2)*COSA(1)*RADIUS)
      XPRIM(7,1)=DBLE(SINA(2)*COSA(1)*RADIUS)
      XPRIM(8,1)=DBLE(SINA(1)*RADIUS)
      IF (IMODE.NE.0) GO TO 4
C
C     STORE NORMAL TO AZIMUTH PLANE AT LAUNCH
      UN(1)=SINA(2)*COSA(3)-COSA(2)*SINA(1)*SINA(3)
      UN(2)=-SINA(2)*SINA(1)*SINA(3)-COSA(2)*COSA(3)
      UN(3)=COSA(1)*SINA(3)
C
C
C     STORE UNIT VECTOR IN THE LAUNCH RADIUS DIRECTION
      UT(1)=COSA(2)*COSA(1)
      UT(2)=SINA(2)*COSA(1)
      UT(3)=SINA(1)
C
    4 COS1 = COSA(1)*SINA(4)-COSA(4)*COSA(3)*SINA(1)
      COS2 = COSA(4)*SINA(3)
C
C     COMPUTE XDOT, YDOT, AND ZDOT.
      XPRIM(3,1)=DBLE(VEL1*(COS1*COSA(2)-COS2*SINA(2))-SNGL(XPRIM(7,1))*
     1REVOLV)
      XPRIM(4,1)=DBLE(VEL1*(COS1*SINA(2)+COS2*COSA(2))+SNGL(XPRIM(6,1))*
     1REVOLV)
      XPRIM(5,1)=DBLE(VEL1*(SINA(1)*SINA(4)+COSA(1)*COSA(3)*COSA(4)))
C
      RETURN
      END
```

```
      SUBROUTINE XOLOAD
C
C         SUBROUTINE XOLOAD DETERMINES WHICH PHASES ARE TO BE OPTIMIZED
C     INTERNALLY BY TERMINATING A PHASE IN THE COAST SUBROUTINE AND
C     WHICH PHASES ARE TO BE OPTIMIZED EXTERNALLY BY ITERATING ON THE
C     PHASE DURATION IN THE NEWTON-RAPHSON SCHEME.
C         THE FIRST PART OF THE PROBLEM IS TO DETERMINE THE EQUATIONS
C     TO BE SATISFIED IN ORDER TO DETERMINE THE OPTIMALLY PHASED
C     VEHICLE.  THESE EQUATIONS ARE OFTEN EVALUATED AT THE INTERMEDIATE
C     PHASING POINTS AND XOLOAD DETERMINES WHERE THE EQUATIONS TO
C     DETERMINE OPTIMUM PHASING ARE TO BE FOUND .  AFTER THIS IS
C     DETERMINED, THE ROUTINE PROPERLY ASSIGNS SOME EQUATIONS TO BE
C     SATISFIED BY COAST AND OTHERS TO THE NEWTON-RAPHSON SCHEME.  THE
C     NUMBER LEFT TO THE NEWTON-RAPHSON SCHEME DETERMINES THE SIZE
C     OF THE ITERATION LOOP REQUIRED.
C
      COMMON /CSTAR/ CMA(1000),CMB(1000)
      DIMENSION FUEL  (6      ),FYD   (5      ),IDATA (6   ,5   )
      DIMENSION JCOAST(6      ),JFINAL(6      ),NOPT  (6        )
      DIMENSION NTB   (4      ),PROP  (6      ),TB    (6        )
      DIMENSION THRUST(6      ),XO    (6   ,5  )
      EQUIVALENCE (CONM  ,CMA(717)),(CONN  ,CMA(718)),(CPA   ,CMA(719))
      EQUIVALENCE (ENERGY,CMA(892)),(FIXDTK,CMB(071)),(FM    ,CMA(715))
      EQUIVALENCE (FUEL  ,CMA(871)),(FYD   ,CMB(036)),(IDATA ,CMB(086))
      EQUIVALENCE (ITER  ,CMB(068)),(ITERAD,CMB(067)),(ITERP ,CMB(069))
      EQUIVALENCE (ITERPD,CMB(060)),(JCOAST,CMB(130)),(JFINAL,CMB(136))
      EQUIVALENCE (LAST  ,CMA(890)),(NCUTE ,CMA(893)),(NFINAL,CMA(879))
      EQUIVALENCE (NOPT  ,CMA(819)),(NOPTA ,CMB(070)),(NTB   ,CMB(116))
      EQUIVALENCE (PROP  ,CMA(849)),(PSIDO ,CMA(882)),(PSIO  ,CMA(881))
      EQUIVALENCE (TB    ,CMA(825)),(THRUST,CMA(831)),(TKICK ,CMB(059))
      EQUIVALENCE (XO    ,CMB(001))
      INTEGER FIXDTK
C
C     ITERPD = 1 FIXED KICK ANGLE
C     ITERPD = 0 OPTIMIZED KICK ANGLE
C
C     COMPUTE CONVERSION FACTOR FOR PRESSURE IN ATMOS
      CPA=9.80665*CONN/CONM**2
C
C     PLACE PSIDO IN THE PERTURBATION ARRAY (XO)
      XO(1,1)=PSIDO
C
C
C     INITIALIZATIONS
      FIXDTK=3
      DO 1  J = 1,6
      JCOAST(J)=0
      JFINAL(J)=0
      FUEL(J)=0.0
      DO 1  K = 1,5
    1 IDATA(J,K)=0
C     FIXDTK = 1 IMPLIES THAT THE KICK ANGLE (TKICK) AND THE BOOSTER
C     PHASE DURATIONS ARE FIXED AND THEN ONLY ONE BOOSTER NEED BE RUN.
      IF(ITERPD.EQ.1.AND.NOPT(1).EQ.0) FIXDTK=1
      NCUTE=0
```

```
      NOPTA=0
      ITERAD=2
      ITER=0
      KA=2
C
C     DETERMINE NUMBER OF PHASES AND SET LAST TO THAT NUMBER.
      DO 2 LAST=1,6
    2 IF(TB(LAST+1).EQ.0.0) GO TO 3
C
C
C     DETERMINE THE NUMBER OF CRITERIA (FIXED TIME AND EQUATIONS)
C     AVAILABLE FOR TERMINATING EACH PHASE.  ALSO DETERMINE WHETHER A
C     PHASE IS POWERED.  NOTE THAT ONE EQUATON IS EXCLUDED FOR THE
C     FIRST OPTIMIZED PHASE.
C     J = 2 REFERS TO FIRST UPPER PHASE
    3 DO 6 J = 1,LAST
C
C     IF PHASE IS UNPOWERED, PUT 1 IN IDATA(J,5).
      IF(THRUST(J).EQ.0.0) IDATA(J,5)=1
C
      IF(NOPT(J).NE.0) GO TO 4
C
C     IF PHASE DURATION IS FIXED, PUT 1 IN IDATA(J,1)
      IDATA(J,1)=1
C
      GO TO 6
C
C     SET NOPTA EQUAL TO FIRST OPTIMIZED PHASE
    4 IF(NOPTA.EQ.0) NOPTA=J
C
      IF(NOPTA.EQ.J) GO TO 6
      IF(PROP(J).EQ.0.0) GO TO 5
C
C     IF PHASE J IS OPTIMIZED AND IT IS NOT THE FIRST OPTIMIZED PHASE
C     AND THE PROPELLANT FRACTION IS NOT EQUAL TO ZERO, SET IDATA(J,2)=1
      IDATA(J,2)=1
C
      GO TO 6
C
C     IF PHASE J IS OPTIMIZED AND IT IS NOT THE FIRST OPTIMIZED PHASE
C     AND THE PROPELLANT FRACTION IS EQUAL TO ZERO, SET IDATA(J-1,3)=1
    5 IDATA(J-1,3)=1
    6 CONTINUE
      DO 7 J = 1,LAST
    7 IDATA(J,4)=IDATA(J,1)+IDATA(J,2)+IDATA(J,3)
C
C
      IF(IDATA(1,1).EQ.1) GO TO 10
      IF(IDATA(1,3).EQ.1) GO TO 8
      ITER=1
      XO(1,3)=TB(1)
      NTB(1)=1
C
      GO TO 10
    8 IF(ITERPD.EQ.0) GO TO 9
C
C     PLACE BOOSTER DURATION IN EXTERNAL ITERATION.  THE PROPELLANT
C     SENSITIVE FRACTION FOR THE SECOND PHASE IS NON-ZERO.  SET ITERAD
C     TO 3 SUCH THAT PSIO IS CHOSEN UTILIZING EQUATION B9 AND B10 FROM
C     PAYLOAD OPTIMIZAION OF MULTISTAGE VEHICLES (NASA TN-3191).  THE
```

```
C     KICK ANGLE IS FIXED.                                              ZXOL0116
      XO(1,2)=TB(1)                                                     ZXOL0117
      NTB(1)=1                                                          ZXOL0118
      KA=1                                                              ZXOL0119
      ITER=1                                                            ZXOL0120
      ITERAD=3                                                          ZXOL0121
C                                                                       ZXOL0122
      GO TO 12                                                          ZXOL0123
C                                                                       ZXOL0124
C     KICK ANGLE IS PLACED IN ITERATION (STATEMENT 11).   THE BOOSTER   ZXOL0125
C     DURATION IS DETERMINED IN START USING THE METHOD UNDER B14 ON     ZXOL0126
C     PAGE 40 OF THE ABOVE MENTIONED REPORT.                            ZXOL0127
    9 ITERAD=1                                                          ZXOL0128
C                                                                       ZXOL0129
      GO TO 11                                                          ZXOL0130
   10 IF(ITERPD.EQ.0) GO TO 11                                          ZXOL0131
C                                                                       ZXOL0132
C                                                                       ZXOL0133
C     PLACE PSIO IN THE EXTERNAL ITERATION.                             ZXOL0134
      XO(1,2)=PSIO                                                      ZXOL0135
C                                                                       ZXOL0136
      GO TO 12                                                          ZXOL0137
C                                                                       ZXOL0138
C     PLACE KICK ANGLE (TKICK) IN THE PERTURBATION ARRAY (XO)           ZXOL0139
   11 XO(1,2)=TKICK                                                     ZXOL0140
C                                                                       ZXOL0141
C     IF ONLY TWO PHASES ARE CONSIDERED, THEN THE NEXT SECTION MAY BE   ZXOL0142
C     SKIPPED.                                                          ZXOL0143
   12 IF(LAST.EQ.2) GO TO 14                                            ZXOL0144
C                                                                       ZXOL0145
      LAST1=LAST-1                                                      ZXOL0146
      DO 13 J=2,LAST1                                                   ZXOL0147
C                                                                       ZXOL0148
C     IF IDATA(J,4) IS NOT EQUAL TO ZERO AND THE JTH. PHASE IS NOT FIXEDZXOL0149
C     THEN AN INTERNAL OPTIMIZATION EQUATION EXISTS AND THE PHASE       ZXOL0150
C     DURATION MAY BE DETERMINED IN COAST (SEE COAST).  OTHERWISE,      ZXOL0151
C     PHASE DURATION MUST BE A PART OF THE EXTERNAL ITERATION.          ZXOL0152
      IF(IDATA(J,4).NE.0) GO TO 13                                      ZXOL0153
C                                                                       ZXOL0154
C     PLACE THOSE PHASE DURATIONS WHICH MAY NOT BE TERMINATED           ZXOL0155
C     INTERNALLY IN THE EXTERNAL ITERATION.                             ZXOL0156
      ITER=ITER+1                                                       ZXOL0157
      K = ITER+KA                                                       ZXOL0158
      XO(1,K)=TB(J)                                                     ZXOL0159
      NTB(ITER)=J                                                       ZXOL0160
   13 CONTINUE                                                          ZXOL0161
C                                                                       ZXOL0162
C                                                                       ZXOL0163
C     AT SOME PHASING POINTS TWO EQUATIONS MUST BE SATISFIED.  WHEN THISZXOL0164
C     OCCURS, THE PHASING POINT MAY BE DETERMINED INTERNALLY WITH ONE OFZXOL0165
C     THESE EQUATONS AND THE OTHER MUST BE SATISFIED AS ONE OF THE FINALZXOL0166
C     CONDITIONS.                                                       ZXOL0167
C     THE JFINAL(J) AND JCOAST(J) ARE FLAGS FOR USE IN FINAL AND COAST  ZXOL0168
C     ENABLING THEM TO EVALUATE THE PROPER EQUATION.                    ZXOL0169
   14 DO 21 J = 1,LAST                                                  ZXOL0170
      IF(IDATA(J,4)-1) 21,15,18                                         ZXOL0171
   15 IF(IDATA(J,1).EQ.1) GO TO 21                                      ZXOL0172
   16 IF(IDATA(J,2).EQ.1) GO TO 17                                      ZXOL0173
      JCOAST(J)=2                                                       ZXOL0174
      GO TO 21                                                          ZXOL0175
```

```
   17 JCOAST(J)=1
      GO TO 21
   18 IF(IDATA(J,3).EQ.1) GO TO 19
      JFINAL(J)=2
      GO TO 20
   19 JFINAL(J)=1
   20 IF(IDATA(J,1).EQ.0) GO TO 16
   21 CONTINUE
C
C     IF THERE IS NO EQUATION TO TERMINATE THE LAST PHASE, THE LAST
C     PHASE IS TERMINATED AT THE PROPER ENERGY (NCUTE = 1)
      IF(IDATA(LAST,4).EQ.0) NCUTE=1
C
C     IF THE OPTIMUM PSIO IS FOUND IN START, THE ITERATION LOOP SIZE
C     (ITERP) IS DECREASED BY ONE.
      ITERP=ITER
      IF(ITERAD.EQ.3) ITERP=ITERP-1
C
      IF(NFINAL.NE.1) GO TO 23
   22 IF(ENERGY.NE.0.0) GO TO 23
C
C     CALCULATE ENERGY
C          FYD(1) RADIUS
C          FYD(2) RADIAL VELOCITY
C          FYD(3) ANGULAR VELOCITY
      ENERGY=(FYD(2)**2+FYD(3)**2*FYD(1)**2)/2.0-FM/FYD(1)
C
   23 IF(JCOAST(LAST).NE.1) RETURN
      JCOAST(LAST)=0
      JFINAL(LAST)=2
      NCUTE=1
      RETURN
      END
```

```
      FUNCTION ZMIN (PERM,N)
C
C          FUNCTION ZMIN COMPUTES ANY VARIABLE ZMIN SUCH THAT ZMIN
C     EQUALS THE SMALLEST MEMBER OF THE FIRST N MEMBERS OF THE
C     PERM ARRAY.
C
      DIMENSION PERM(6),TEMP(6)
      DO 1 J = 1,N
    1 TEMP(J)=PERM(J)
      DO 2 J = 2,N
      IF(TEMP(J-1).GE.TEMP(J)) GO TO 2
      TEMP(J)=TEMP(J-1)
    2 CONTINUE
      ZMIN=TEMP(N)
      RETURN
      END
```

REFERENCES

1. Teren, Fred; and Spurlock, Omer F.: Payload Optimization of Multistage Launch Vehicles. NASA TN D-3191, 1966.

2. Willwerth, Robert E., Jr.; Rosenbaum, Richard C.; and Chuck, Wong: PRESTO Program for Rapid Earth-to-Space Trajectory Optimization. NASA CR-158, 1965.

3. Strack, William C.; Dobson, Wilbur F.; and Huff, Vearl, N: The N-Body Code - A General Fortran Code for Solution of Problems in Space Mechanics by Numerical Methods. NASA TN D-1455, 1963.

4. Campbell, K. E.: Trajectory Data Book, AC-4. Rep. GDA63-6439-1, General Dynamics/Convair, Sept. 15, 1964.

5. Turner, Don N.; and Huff, Vearl N.: An Input Routine Using Arithmetic Statements for the IBM 704 Digital Computer. NASA TN D-1092, 1961.

6. Dellner, Lois T.: A Set of FORTRAN IV Subroutines for Generating Printed Plots. NASA TM X-1419, 1967.

NASA-Langley, 1968 — 30

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION
WASHINGTON, D.C. 20546

OFFICIAL BUSINESS

FIRST CLASS MAIL

POSTAGE AND FEES P
NATIONAL AERONAUTIC
SPACE ADMINISTRATI

04U 001 55 51 3DS 68194 00903
AIR FORCE WEAPONS LABORATORY/AFWL/
KIRTLAND AIR FORCE BASE, NEW MEXICO 87117

ATT MISS MADELINE F. CANOVA, CHIEF TECHNI
LIBRARY /WLIL/

POSTMASTER: If Undeliverable (Sectio
Postal Manual) Do Not

"The aeronautical and space activities of the United States shall be conducted so as to contribute . . . to the expansion of human knowledge of phenomena in the atmosphere and space. The Administration shall provide for the widest practicable and appropriate dissemination of information concerning its activities and the results thereof."

—NATIONAL AERONAUTICS AND SPACE ACT OF 1958

NASA SCIENTIFIC AND TECHNICAL PUBLICATIONS

TECHNICAL REPORTS: Scientific and technical information considered important, complete, and a lasting contribution to existing knowledge.

TECHNICAL NOTES: Information less broad in scope but nevertheless of importance as a contribution to existing knowledge.

TECHNICAL MEMORANDUMS: Information receiving limited distribution because of preliminary data, security classification, or other reasons.

CONTRACTOR REPORTS: Scientific and technical information generated under a NASA contract or grant and considered an important contribution to existing knowledge.

TECHNICAL TRANSLATIONS: Information published in a foreign language considered to merit NASA distribution in English.

SPECIAL PUBLICATIONS: Information derived from or of value to NASA activities. Publications include conference proceedings, monographs, data compilations, handbooks, sourcebooks, and special bibliographies.

TECHNOLOGY UTILIZATION PUBLICATIONS: Information on technology used by NASA that may be of particular interest in commercial and other non-aerospace applications. Publications include Tech Briefs, Technology Utilization Reports and Notes, and Technology Surveys.

Details on the availability of these publications may be obtained from:

SCIENTIFIC AND TECHNICAL INFORMATION DIVISION

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION

Washington, D.C. 20546